Hierarchical Blockmodelling for Knowledge Graphs

Marcin Pietrasik [a,b,*], Marek Reformat [a,c] and Anna Wilbik [b]
[a] *Department of Electrical and Computer Engineering, University of Alberta, Edmonton, Canada*
[b] *Department of Advanced Computing Science, Maastricht University, Maastricht, The Netherlands*
[c] *Information Technology Institute, University of Social Sciences, Łódź, Poland*
E-mails: marcin.pietrasik@maastrichtuniversity.nl, reformat@ualberta.ca, a.wilbik@maastrichtuniversity.nl

Abstract. In this paper, we investigate the use of probabilistic graphical models, specifically stochastic blockmodels, for the purpose of hierarchical entity clustering on knowledge graphs. These models, seldom used in the Semantic Web community, decompose a graph into a set of probability distributions. The parameters of these distributions are then inferred allowing for their subsequent sampling to generate a random graph. In a non-parametric setting, this allows for the induction of hierarchical clusterings without prior constraints on the hierarchy's structure. Specifically, this is achieved by the integration of the Nested Chinese Restaurant Process and the Stick Breaking Process into the generative model. In this regard, we propose a model leveraging such integration and derive a collapsed Gibbs sampling scheme for its inference. To aid in understanding, we describe the steps in this derivation and provide an implementation for the sampler. We evaluate our model on synthetic and real-world datasets and quantitatively compare against benchmark models. We further evaluate our results qualitatively and find that our model is capable of inducing coherent cluster hierarchies in small scale settings. The work presented in this paper provides the first step for the further application of stochastic blockmodels for knowledge graphs on a larger scale. We conclude the paper with potential avenues for future work on more scalable inference schemes.

Keywords: knowledge graphs, stochastic blockmodels, hierarchical clustering

1 Introduction

In recent years, using graph structures to model and store data has been garnering an increasing amount of attention among practitioners in sectors ranging from academia to government to industry. Indeed by some measures [1, 2], graph database management systems are the fastest growing database type over the past decade. One of the more obvious manifestations of this rise is the recent growth of large scale public graph databases such as DBpedia [3], YAGO [4], and Wikidata [5]. The last of these, for instance, contains just over 100 million entities as of 2024, a near seven fold increase over its count in 2014. The open access to such amounts of graph data has spurred on its use in research related to the Semantic Web, artificial intelligence, and computer science broadly. One field of research which has received considerable attention is that of mathematically modelling the underlying graph structure that emerges when a knowledge base is populated by information. The modelling of this structure – which we refer to as the knowledge graph – proves useful in its application to solve downstream problems such as link prediction, entity clustering, and hierarchy induction. The last two of these provided the impetus for our work.

Entity clustering refers to the task of grouping together entities in a knowledge graph which share similar properties. The measure by which entities are judged to be similar varies and is one of the key considerations when

[*] Corresponding author.

devising an approach to their clustering. Obtaining an entity clustering allows for the discovery of structures which are implicit in the knowledge graph and provides insight into the number and types of categories which exist in the data. The process operates on unlabelled data and is therefore a type of unsupervised learning. As such, it is one of the first and most useful operations applied to a knowledge graph when performing exploratory analysis. Another important unsupervised learning task is that of hierarchy induction. The clearest example of a knowledge graph hierarchy is the class taxonomy which organizes a knowledge graph's classes through superclass-subclass relations. The task of inducing such a taxonomy merely amounts to learning how the classes are organized hierarchically in the knowledge graph. Similarly, hierarchical clustering of a knowledge graph's entities extends the clustering task described earlier by imposing a hierarchical organization to the clusters themselves. This allows not only to discover which entities are semantically similar as per the clustering but also how entities relate to one another hierarchically. The motivating factors behind learning knowledge graph hierarchies are various. Perhaps the simplest is that hierarchical structures organize data in a way that is highly intuitive and interpretable to humans. For instance, a hierarchical clustering of knowledge graph entities makes it apparent which entities constitute the broadest concepts in the knowledge graph and how they relate to their descendants. Similarly, a taxonomy of classes reveals implicit relations between entities through its transitive properties. Put plainly, hierarchies induced from knowledge graphs are useful because they are easy to understand. Indeed, the most widely used knowledge bases – such as the aforementioned DBpedia, YAGO, and Wikidata – are organized by hierarchical structures, namely trees and directed acyclic graphs. That is to say, these knowledge graphs are hierarchical at their core. Furthermore, hierarchies are used as components of larger systems to solve common tasks related to knowledge graphs. For instance, hierarchies are used in learning knowledge graph embeddings, both explicitly as an input feature of the model [6] and implicitly as a byproduct of the embedding process [7]. As embedding is one of the most common problems in the knowledge graph community, learning accurate hierarchies is therefore desirable.

In this regard, our work proposes a generative model for knowledge graphs which induces a clustering of entities and organizes it hierarchically. Our approach belongs to a class of probabilistic graphical models called stochastic blockmodels. In broad strokes, these models operate by decomposing a knowledge graph into a set of probability distributions which are then sampled from to generate the knowledge graph. As a byproduct of this sampling process, a hierarchical clustering of knowledge graph entities is induced. To the best of our knowledge, our approach is the first to apply stochastic blockmodels to knowledge graphs and one of a very few probabilistic graphical models to be used for the purpose of knowledge graph hierarchy induction. To highlight this, we position our work in the context of existing stochastic blockmodels and hierarchy induction methods in Section 2 and provide a gentle introduction for their understanding in Section 3. The formal definition of our model that follows in Section 4 results in a joint distribution which is intractable for exact inference. The parameters for our model must therefore be approximated using collapsed Gibbs sampling. To this end, we provide the full derivation of sampling equations as well as the marginalization of collapsed variables. Additional information to supplement Section 4 may be found in Appendices A through D. Section 6 concludes the paper by summarizing its contributions and providing avenues for future work.

2 Related Work

Our proposed model lies at the intersection of two areas in artificial intelligence which deal with modelling graph data: stochastic blockmodelling and hierarchy induction. Due to the limited overlap of these fields, we provide separate summaries of related works for each.

2.1 Stochastic Blockmodels

Stochastic blockmodels are a class of probabilistic graphical models used for generating random graphs with roots in the fields of social science and mathematics. First proposed in 1983 by Holland et al. [8] for modelling social networks, they have expanded their utility to fields such as biochemistry [9], education [10], and artificial intelligence [11–13] among others. In simplest terms, stochastic blockmodels are a type of Bayesian non-parametric graph partition model in that their approach relies on grouping graph entities together via partitions – often referred to as blocks – which share similar structural properties. The generative process by which this partitioning occurs is realized by sampling from a set of probability distributions, giving rise to the stochasticity of stochastic blockmodels. The learning process is then to infer the parameters of these distributions using a Bayesian inference scheme. We provide a technical introduction to stochastic blockmodels in the subsequent section.

The seminal work in this area is the Stochastic Blockmodel [14] which partitions entities into a fixed number of communities and models the interactions between them as those of their communities. Community relations are modelled via a community relations matrix which assigns a degree to all pairwise interactions between the communities in the model. This idea was extended to the infinite case allowing for an a priori unspecified number of communities via the Chinese restaurant process [15] in the Infinite Relational Model [16] and its recent hierarchical counterpart the Hierarchical Infinite Relational Model [17]. A variant which relaxes the notion of community membership to allow for entities belonging to multiple communities is the aptly named Mixed Membership Stochastic Blockmodel [11]. By allowing for mixed membership, the model is better able to capture entities whose belonging to a community is not crisp. For instance, the belonging of tomatoes to the community of fruits is not perfect since it can be considered a vegetable in certain contexts such as in cooking. This idea was generalized to the infinite case in the Dynamic Infinite Mixed Membership Stochastic Blockmodel [18] and the hierarchical case in the Multiscale Community Blockmodel [12]. The latter of these two is closely related to our model and receives more attention later in the paper. All of the aforementioned models, however, operate on graphs wherein entities are related to one another through the same type of edge, making them unsuitable for application to knowledge graphs without modification.

The underlying structure of a knowledge graph is that of a multilayer graph wherein entities interact with one another through different types of relations, represented as different types of edges in the graph. These relations may be thought of as separate layers of graphs which share the same entities. Multilayer graphs have also received considerable attention in stochastic blockmodelling. Perhaps the simplest approach is to aggregate the layers in the multilayer graph to a single layer before applying a conventional blockmodelling approach as was done in Berlingerio et al. [19]. A closely related approach is to model each layer in the graph independently as done in Barigozzi et al. [20] and aggregate the results afterwards. These approaches offer limited success as they don't capture the interlayer dependencies in the multilayer graph and treat each layer as equally valuable in its content during modelling, as pointed out by Paul and Chen [21]. To remedy this, the authors propose a multilayer extension of the aforementioned Stochastic Block Model, aptly named the Multi-Layer Stochastic Blockmodel, which modifies the original community relations matrix to a community relations tensor to account for graph multilayeredness. Analogously, a multilayer extension for the Mixed Membership Stochastic Blockmodel was proposed by De Bacco et al. [22]. Finally, the Multilayer Neural Blockmodel [23] was proposed recently as a way to marry neural networks with the probabilistic approach of stochastic blockmodels for modelling multilayer graphs. A comprehensive review of stochastic blockmodels and their applications is provided by Lee and Wilkinson [24].

2.2 Hierarchy Induction Models

In the context of our work, hierarchy induction refers to the discovery of hierarchical structures which are implicit and otherwise unexpressed in a knowledge graph. One concrete way this task is formulated is as that of learning subsumption axioms for classes in a knowledge graph, thereby discovering a hierarchical organization of a knowledge graph's entities. To this end, Statistical Schema Induction [25] uses association rule mining on a knowledge graph's transaction table to generate subsumption axioms with support and confidence values which are then used as the basis for a greedy algorithm for constructing an ontology. SMICT [26] transforms a knowledge graph into a tuple structure wherein entities are annotated by tags and applies a greedy algorithm to learn a taxonomy of classes. This method was extended to perform hierarchical clustering using the Jaccard coefficient [27]. In general, by transforming a knowledge graph to a tuple structure, various [28–30] methods in the area of tag hierarchy induction can be leveraged. In a related approach, Chen and Reformat [31] derive a similarity matrix from a knowledge graph's tuple structure which serves as the clustering metric for hierarchical agglomerative clustering. Mohamed [32] takes a similar approach wherein subjects which are described by the same tag pairs are assigned to the same groups. The similarity between these groups is then calculated to construct a hierarchy. In a method which bears similarity to our own, Zhang et al. [13] use a non-parametric Bayesian approach to induce a hierarchy of topic communities. Despite a similar statistic framework and inference scheme, the hierarchy induced by this work differs significantly from our own. For instance, relations between communities are not modelled and entities are never explicitly assigned to communities. Along similar lines is GMMSchema [33] which uses a Gaussian mixture model to generate a schema graph which can be viewed as a hierarchical abstraction of the original knowledge graph.

Another common approach to learning hierarchies from knowledge graphs is via an intermediate representation which lends itself well to existing hierarchy induction methods. To this end, knowledge graph embedding is oftentimes leveraged. This process involves learning a mapping from the discrete knowledge graph to a continuous vector space. The vector representation may then serve as the input to machine and deep learning methods for hierarchy learning. Translation based methods such as the seminal TransE [34] and its extensions [35–37] treat relations in a knowledge graph as translations between entities. Additive in nature, they operate on the intuition that embeddings of subjects and objects should be proximal when translated by the relation of a valid triple. These embeddings are learned by minimizing an objective function using an optimization method such as stochastic gradient descent. Bilinear methods [38–41] operate on the binary adjacency tensor of the knowledge graph and factorize entities and relations into vectors and matrices. Triples are then modelled as their resulting product. These methods tend to perform well on measures of performance compared to translation based methods but suffer from higher training complexity. Deep learning models have also been proposed in the context of knowledge graph embeddings. For instance, the Relational Graph Convolution Network [42] leverages graph convolutions to learn neighbourhood information of entities, thereby explicitly incorporating structural information into its modelling. Another widely used deep approach, ConvE [43], stacks subject and predicate embeddings as a matrix and convolves over them in two dimensions using a neural framework. This approach was extended in ConvKB [44] which incorpoates objects into the convolution process and CapsE [45] which uses a similar architecture with capsule layers to yield scores for triples. A recent and comprehensive comparative analysis of various embedding methods may be found in [46].

Having obtained an embedded representation of a knowledge graph, hierarchical clustering methods can be applied to induce a hierarchy. For instance RESCAL [38], a bilinear embedding method, was used in conjunction with OPTICS [47], a density based hierarchical clustering algorithm, in Nickel et al. [48] to obtain a hierarchical clustering of entities. They found that such an approach achieves more coherent results for concepts which appear at the top of the hierarchy, largely due to data sparsity for descendant concepts. Along similar lines, TIEmb [49] generates embeddings using RDF2Vec [50], an embedding method based on the skip-gram language model [51], before learning a hierarchical structure based on the proximities of class centroids in the embedded space. The same embedding approach was used in Martel and Zouaq [52] wherein the embeddings were then clustered using hierarchical agglomerative clustering and assigned types. This type of clustering was used in the field of cybersecurity in Ding et al. [53] wherein a bag-of-words representation of a knowledge graph served as input. Compared with the aforementioned subsumption axiom induction methods which rely largely on frequencies and co-occurrences between type classes, embedding based approaches typically embed an entire knowledge graph, thus leveraging a larger and much richer body of information. As such, when compared to subsumption axiom methods, one would expect embedding based methods to be more robust to datasets poorly structured in terms of their type information. To the best of our knowledge, there is yet to be an analysis performed which compares the two approaches.

3 Preliminaries

Before describing the details of our proposed model, we provide a basic overview of several concepts necessary for its understanding. These concepts are described only insofar as to provide readers with the foundation on which the explanation of our model can be built. We implore readers unfamiliar with knowledge graphs or Bayesian nonparametrics to follow the relevant citations provided in each of the subsequent subsections. To aid in readability we use the following conventions in our notation: lowercase italic Latin letters for iterators and indexers; uppercase italic Latin letters for scalar variables; lowercase boldface Latin letters for vectors; uppercase boldface Latin letters for matrices and tensors; uppercase stylized Latin letters for sets; lowercase Greek letters for hyperparameters; and uppercase Greek letters for functions.

3.1 Knowledge Graphs

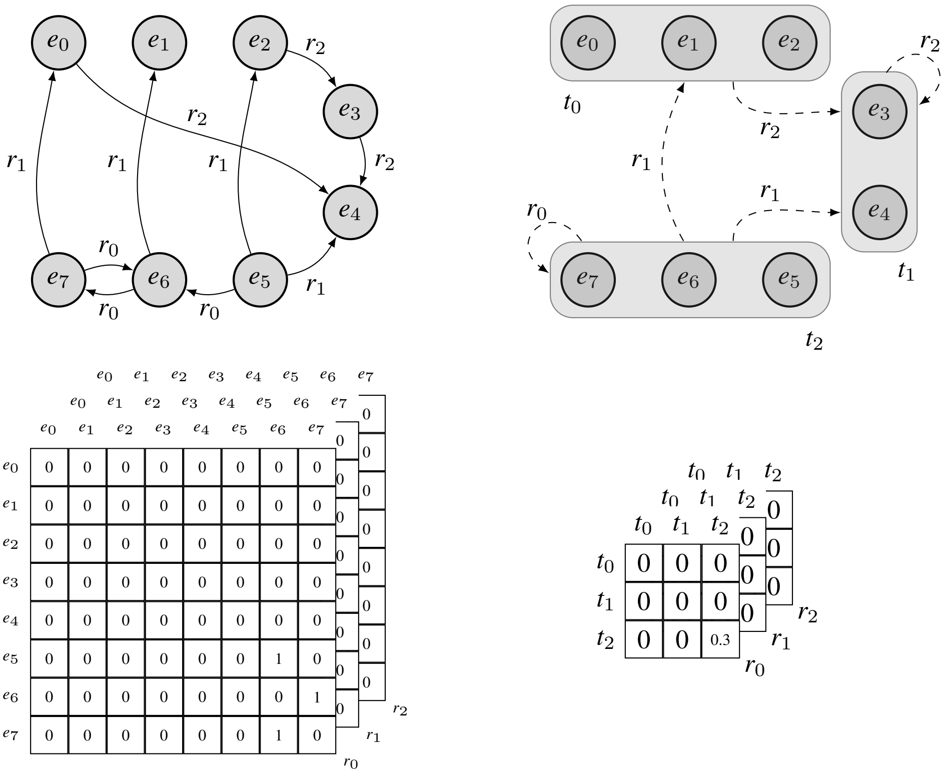
We refer to Hogan et al. [54] for their definition of knowledge graphs as *"a graph of data intended to accumulate and convey knowledge of the real world, whose nodes represent entities of interest and whose edges represent potentially different relations between these entities."* Concretely, information is stored as a collection of triples wherein each triple relates a subject entity, e_i, to an object entity, e_j, via a predicate, p_r. Formally, we define a knowledge graph, $\mathcal{G}$, as a set such that $\mathcal{G} = \{\langle e_i, r_p, e_j \rangle \in \mathcal{E} \times \mathcal{R} \times \mathcal{E}\}$ where $\langle e_i, r_p, e_j \rangle$ is a triple, $\mathcal{E}$ is the set of entities in $\mathcal{G}$, and $\mathcal{R}$ is the set of predicates in $\mathcal{G}$. When put together, the triples form a directed graph with nodes corresponding to entities and edges corresponding to predicates. Each triple in a knowledge graph describes one piece of information or fact. For instance, ⟨`Henry Ford, occupation, Engineer`⟩ relates the subject `Henry Ford` to the object `Engineer` through the predicate `occupation` and states, in plain English, that Henry Ford's occupation is being an engineer. Notice that this definition of knowledge graphs allows for cycles and entity self-relations to exist. This is made clear when analyzing a knowledge graph's binary adjacency tensor which may be asymmetric and containing non-zero values in its main diagonal. Knowledge graphs are oftentimes represented in their tensor form as it allows for easier numerical operation and thus opens the door to various tools and methods in artificial intelligence. A binary adjacency tensor is obtained from a knowledge graph by ordering its entities and predicates along an $|\mathcal{E}| \times |\mathcal{E}| \times |\mathcal{R}|$ tensor, $\mathbf{G}$, that takes on values $g_{ijr} = 1$ if there exists a triple in $\mathcal{G}$ from entity e_i to entity e_j on predicate r_p and $g_{ijr} = 0$ otherwise. This representation is used in stochastic blockmodelling and is the one we will use in this paper henceforth. The left half of Figure 1 depicts a simple knowledge graph along with its adjacency tensor representation. A comprehensive introduction to knowledge graphs is provided by Gutierrez and Sequeda [55].

3.2 Stochastic Blockmodels

Stochastic blockmodels are a heterogeneous collection of generative models united in their adoption of two characteristics: stochasticity in the generative process and the partitioning of nodes into communities. Describing them by referring to a concrete instance is thus bound to include definitions which do not apply to all members of the class. With this in mind, our introduction to stochastic blockmodels draws on their key characteristics to motivate a toy stochastic blockmodel for generating a knowledge graph. All stochastic blockmodels are defined by a set of probability distributions from which samples are obtained to generate the adjacency tensor of the knowledge graph, $\mathbf{G}$, indexed by g_{ijr} where the subscript $_{ijr}$ indicates the value associated with the triple from e_i to e_j on predicate p_r. In order to perform this generation, the knowledge graph's entities must first be assigned to one of the model's communities. This is done by sampling the model's variables responsible for this assignment. Let $\mathbf{A}$ be a tensor representing these variables with a corresponding hyperparameter α responsible for parameterizing their prior distribution. In stochastic blockmodels, the probability of an interaction between two entities is modelled as the degree of interaction between their respective communities. It is necessary, therefore, to capture these community relations by sampling their corresponding model variables. Let $\mathbf{B}$ be a tensor representing this subset of variables with a prior hyperparameter β. The joint distribution of this model is obtained by applying the chain rule of probability as follows:

$$\mathbb{P}(\mathbf{G}, \mathbf{A}, \mathbf{B} \mid \alpha, \beta) = \prod_{g_{ijr} \in \mathbf{G}} \mathbb{P}(g_{ijr} | \mathbf{A}_{ijr}, \mathbf{B}_{ijr}, \alpha, \beta) \mathbb{P}(\mathbf{B}_{ijr} \mid \mathbf{A}_{ijr}, \beta) \mathbb{P}(\mathbf{A}_{ijr} \mid \alpha) \tag{1}$$

Where $\mathbf{A}_{ijr}$ and $\mathbf{B}_{ijr}$ indicate the latent variables in $\mathbf{A}$ and $\mathbf{B}$ associated with sampling g_{ijr}. Notice that the probability of drawing a value in the knowledge graph's adjacency tensor, $\mathbb{P}(g_{ijr} | \mathbf{A}_{ijr}, \mathbf{B}_{ijr}, \alpha, \beta)$, is conditioned on $\mathbf{A}$ and $\mathbf{B}$. Thus, in order to generate the knowledge graph, it is necessary to first infer the values of $\mathbf{A}$ and $\mathbf{B}$. This inference process is analogous to the training phase of other machine and deep learning models. In most cases, the solution is intractable for exact inference and must be approximated using an inference scheme. Perhaps the simplest inference scheme used in stochastic blockmodelling is Gibbs sampling, a Markov chain Monte Carlo method which can be used for sampling from a joint distribution. Gibbs sampling approximates this distribution by iteratively sampling from its variables' full conditional distributions. This iterative sampling creates a Markov chain of samples wherein



	e_0	e_1	e_2	e_3	e_4	e_5	e_6	e_7
e_0	0	0	0	0	0	0	0	0
e_1	0	0	0	0	0	0	0	0
e_2	0	0	0	0	0	0	0	0
e_3	0	0	0	0	0	0	0	0
e_4	0	0	0	0	0	0	0	0
e_5	0	0	0	0	0	0	1	0
e_6	0	0	0	0	0	0	0	1
e_7	0	0	0	0	0	0	1	0

	t_0	t_1	t_2
t_0	0	0	0
t_1	0	0	0
t_2	0	0	0.3

Fig. 1. Toy example of a knowledge graph and how it may be modelled by a stochastic blockmodel. Starting from top left quadrant and proceeding clockwise: graphical representation of a knowledge graph with entities e_0 through e_7 and predicates r_0 through r_2; graphical representation of aforementioned knowledge graph as modelled by a stochastic blockmodel with communities t_0 through t_2; potential community relations tensor induced by stochastic blockmodel; adjacency tensor of knowledge graph above it.

its stationary distribution approximates the joint distribution of the model. Continuing the example above, to infer the blockmodel's parameters for g_{ijr}, namely $\mathbf{A}_{ijr}$ and $\mathbf{B}_{ijr}$, inference is performed on their conditional distributions $\mathbb{P}(\mathbf{A}_{ijr} \mid \mathbf{G}, \mathbf{B}, \alpha)$ and $\mathbb{P}(\mathbf{B}_{ijr} \mid \mathbf{G}, \mathbf{A}, \beta)$, respectively. We apply Bayes' theorem to obtain these distributions. Recall that by this theorem the posterior distribution is proportional to the product of the likelihood and the prior. We can therefore express the conditionals of $\mathbf{A}_{ijr}$ and $\mathbf{B}_{ijr}$ as follows:

$$\mathbb{P}(\mathbf{A}_{ijr} \mid \mathbf{G}, \mathbf{B}, \alpha) \propto \mathbb{P}(\mathbf{G} \mid \mathbf{A}_{ijr}, \mathbf{B})\mathbb{P}(\mathbf{A} \mid \alpha) \tag{2}$$

$$\mathbb{P}(\mathbf{B}_{ijr} \mid \mathbf{G}, \mathbf{A}, \beta) \propto \mathbb{P}(\mathbf{G} \mid \mathbf{B}_{ijr}, \mathbf{A})\mathbb{P}(\mathbf{B} \mid \beta) \tag{3}$$

Where $\mathbb{P}(\mathbf{G} \mid \mathbf{A}, \mathbf{B})$ and $\mathbb{P}(\mathbf{G} \mid \mathbf{B}, \mathbf{A})$ are the likelihoods, and $\mathbb{P}(\mathbf{A}_{ijr} \mid \alpha)$ and $\mathbb{P}(\mathbf{B}_{ijr} \mid \beta)$ are the priors of $\mathbf{A}_{ijr}$ and $\mathbf{B}_{ijr}$, respectively. The likelihood may be understood as the of chance observing the data given the model parameters. In Equations 2 and 3, it is the likelihood of drawing $\mathbf{G}$ from our model with parameters $\mathbf{A}$ and $\mathbf{B}$. The prior represents the assumptions about a variable before any data is taken into account. They are oftentimes chosen in order to leverage a conjugacy with their dependant variables. Priors are parameterized by hyperparameters which must be specified a priori. The choice of these hyperparameters influences the density of the prior and can thus change the output of the model. Gibbs sampling draws from the variables' full conditional distributions iteratively for a predetermined number of iterations, *iters*. To highlight this, the superscript *iter* is added to denote the value of a variable at the corresponding iteration. The Gibbs sampling process may be summarized as follows:

1. Initialize $\mathbf{A}^0_{ijr}$ and $\mathbf{B}^0_{ijr}$ for each $g_{ijr} \in \mathbf{G}$
2. For iteration *iter* in $1, 2, ..., iters$

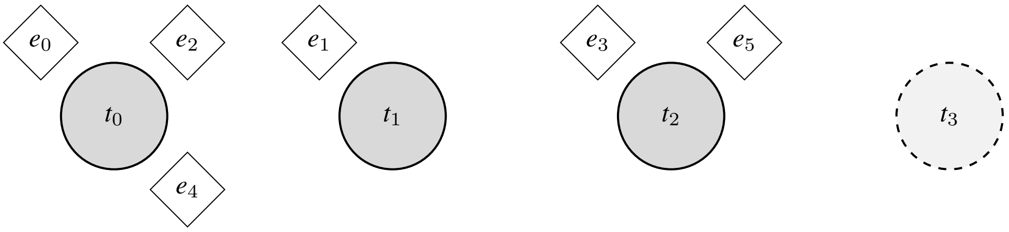


Fig. 2. Toy example of the CRP after sitting patrons e_0 through e_5. Tables t_0 through t_2 are occupied and table t_3 is the next unoccupied table. We illustrate Equation 4 by calculating the probabilities of sitting patron e_6 at tables t_0 and t_3: $\mathbb{P}(e_6 = t_0) = \frac{3}{6+\gamma}$ and $\mathbb{P}(e_6 = t_3) = \frac{\gamma}{6+\gamma}$.

(a) Sample $\mathbf{A}_{ijr}^{iter} \sim \mathbb{P}(\mathbf{A}_{ijr}^{iter} \mid \mathbf{G}, \mathbf{B}^{iter-1}, \alpha)$ for each $g_{ijr} \in \mathbf{G}$ using Equation 2
(b) Sample $\mathbf{B}_{ijr}^{iter} \sim \mathbb{P}(\mathbf{B}_{ijr}^{iter} \mid \mathbf{G}, \mathbf{A}^{iter-1}, \alpha)$ for each $g_{ijr} \in \mathbf{G}$ using Equation 3

In step 1, variables can be initialized by sampling from their prior distributions or specified explicitly if a priori evidence to suggest their true values exists. Step 2 depicts the iterative sampling of model variables from their full conditionals. We note that samples obtained early in this process may be drawn from a distribution distant to that of the desired stationary distribution. As such it is necessary to discard the samples obtained before this distribution has been reached. This process is commonly referred to as burning in the Gibbs sampler and the number of discarded iterations as the burn in iterations. Furthermore, as successive samples in this process are autocorrelated, there may be a lag period applied in obtaining results such that samples in during the lag period are also discarded. Thus, if our toy example performs 1000 iterations with a burn in of 900 and a lag of 10, only 9 samples will be obtained as the output of the Gibbs sampler. These 9 samples are then aggregated over to account for the stochasticity in sampling from the posterior and arrive at a final result. The process by which these samples are aggregated are model specific and may be as simple as merely taking the sampled mode. An introduction to Gibbs sampling and related sampling schemes is covered by Mackay [56] and a thorough discussion of stochastic blockmodels along with their concrete examples is provided by Abbe [57].

3.3 The Chinese Restaurant Process

The Chinese restaurant process (CRP) [15] is a discrete stochastic process that yields a probability distribution in accordance with the preferential attachment principle. In this view, it is both a Dirichlet process [58] as it generates a probability distribution and a preferential attachment process [59] as the distribution is generated such that probabilities are proportional to past draws. The process is explained through a metaphor of sitting patrons at a Chinese restaurant. Consider this restaurant as containing an infinite number of tables with each table having the capacity to seat an infinite number of patrons. Patrons are seated sequentially, such that the first patron is seated at the first table and every subsequent patron may be seated at an occupied table or the first unoccupied table. The probability of being seated at an occupied table is proportional to the number of patrons already seated at it. This process is illustrated through the toy example in Figure 2 which shows a potential state of the CRP after sitting six patrons along with the sample probabilities of sitting the seventh. Formally, the probability of seating patron e_i[1] at a table t_m in a restaurant where $\mathcal{T}_i$ is the set of occupied tables when patron e_i arrives is:

$$\mathbb{P}(e_i = t_m \mid e_0, e_1, ..., e_{i-1}, \gamma) = \begin{cases} \dfrac{\#_i^m}{i+\gamma} & t_m \in \mathcal{T}_i \\ \dfrac{\gamma}{i+\gamma} & t_m \notin \mathcal{T}_i \end{cases} \tag{4}$$

Here, $\#_i^m$ is the number of patrons seated at table t_m when patron e_i arrives and $\gamma > 0$ is a hyperparameter of the CRP responsible for controlling the probability that an incoming patron is seated at an unoccupied table such

[1] We shall see later that patrons in this analogy are equivalent to entities in our model, hence the double use of e to represent entities and patrons. This same principle applies to the use of t to represent both tables and communities.

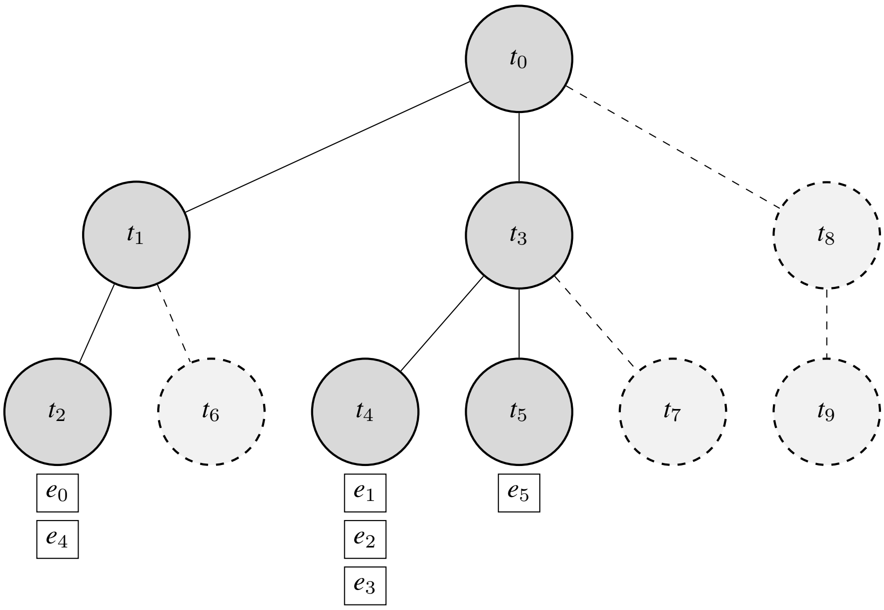


Fig. 3. Toy example of a nCRP truncated to a depth of $L = 2$ after assigning patrons e_0 through e_5. Solid lines indicate paths which have been taken by patrons and thus exist in the tree whereas dashed lines indicate potential paths. We illustrate Equation 6 by calculating the probability of a patron taking a path through communities t_2 and t_9: $\mathbb{P}(e_6 = t_2) = (\frac{2}{2+\gamma})(\frac{2}{6+\gamma})$ and $\mathbb{P}(e_6 = t_9) = \frac{\gamma}{6+\gamma}$.

that increasing γ increases this probability. Thus, increasing γ values will yield results with an increasing number of occupied tables. Specifically, the expected number of occupied tables grows logarithmically with respect to the number of seated patrons:

$$\mathbb{E}\left[\sum_{t_m \in \mathcal{T}_i} \mathbb{I}(\#_i^m > 0) \mid \gamma\right] = \mathcal{O}(\gamma \log i) \tag{5}$$

Where $\mathbb{I}$ is the indicator function which returns 1 if the condition is met and 0 otherwise. Big-O notation is leveraged with $\mathcal{O}$ to indicate the asymptotic upper bound of the expectation. This principle becomes relevant when controlling the branching factor of the induced tree as we will see later on. The realization of the CRP yields a partition of patrons over the infinitely many tables in the restaurant. If we consider each table to be a community, we can leverage this process to obtain a probability distribution over an infinite number of communities. Indeed this is the main utility of the CRP, namely to serve as a conjugate prior to infinite non-parametric discrete distributions. While this approach allows for the modelling of flat communities, it does not account for hierarchical relations between them. To remedy this, the CRP must be extended to its nested variant.

3.4 The Nested Chinese Restaurant Process

The nested Chinese restaurant process (nCRP) [60, 61] is an extension of the CRP formulated to account for hierarchical relations between the generated communities. The realization of this process is an infinitely deep and infinitely branching tree of communities defined by a set of paths, $\mathcal{P}$, taken from the root community to a leaf community. In principle, the tree is unbounded in depth, however we limit our discussion to a nCRP bounded to a depth of L. As in the case of the CRP, the allocation of paths along the tree is consistent with the preferential attachment principle. The tree is generated stochastically by sampling a path at each level in the tree via the CRP such that drawing a table is analogous to taking a path at that level. To extend the metaphor of seating patrons at a Chinese restaurant, consider the scenario of an infinite number of restaurants with an infinite number of infinite seat tables. When patrons are seated at these restaurants they are not served food but rather a table specific reference to another restaurant to which they must go. One of these restaurants is designated a root restaurant with no reference and all other restaurants are referenced exactly once. The seating of patrons at these restaurants is performed as in the CRP. We can see how realizing this process yields a tree by examining the paths taken by patrons. They first arrive at the root restaurant before being sent off to one of the root restaurant's descendant restaurants. At this restaurant the patron is sent off to another descendant restaurant and this process is repeated until L restaurants have been visited in the bounded case. The paths taken by patrons generate the tree as illustrated in the toy example in Figure 3. As before, we extend this analogy of patrons and tables to entities and communities, respectively. Thus, when drawing path $\mathbf{p}_i$ for entity e_i, the process starts by initializing the path at the top level to the root community, namely $p_i^0 = t_0$ where the superscript in path $\mathbf{p}_i$ indexes into the path vector to obtain the community at the corresponding level and t_0 is the root community. The process then continues by drawing a descendent community according to the CRP. Recall that this draw results in a community which either has or has not been visited before by a previous entity. The latter case corresponds to branching out a new path in the tree at the descendant level. This process is repeated L times at which point the specified depth has been reached. We can formalize this process by extending the previously defined notation. Specifically, let $\mathcal{T}_i$ be the set of communities in the tree before entity e_i has its path sampled and $\mathcal{C}_i^q$ be the set of children communities for community t_q at this time as well. The sampling process is then expressed as follows: when entity e_i arrives at community t_q on the $(l-1)^{\text{th}}$ level in the tree, the probability of selecting an existing community, $p_i^l \in \mathcal{C}_i^q$ or creating a new community, $p_i^l \notin \mathcal{T}_i$, is:

$$\mathbb{P}(p_i^l = t_c \mid \mathbf{p}_0, \mathbf{p}_1, ..., \mathbf{p}_{i-1}, \mathbf{p}_i^{1:\,l-1}, \gamma) = \begin{cases} \dfrac{\#_i^{t_c}}{\#_i^{t_q} + \gamma} & t_c \in \mathcal{C}_i^c \\ \dfrac{\gamma}{\#_i^{t_q} + \gamma} & t_c \notin \mathcal{T}_i \end{cases} \tag{6}$$

Where $\#_i^{t_q}$ and $\#_i^{t_c}$ is the number of entities that have passed through communities t_q and t_c before entity e_i started its path. The superscript in $\mathbf{p}_i^{1:l-1}$ indicates that the probability distribution for sampling p_i^l is conditioned on the path taken by entity e_i up until level l. The hyperparameter γ serves a similar function as in the CRP, namely controlling the branching factor of the tree such that higher γ values yield trees with more branches. The use of the CRP in the path decision process ensures that probability mass will be pulled towards drawing paths which have been more frequently drawn before. The resulting distribution allows us to use the nCRP as a non-parametric prior over a tree structure in our model. In drawing paths, we not only generate a hierarchy but also define a subset of communities to which an entity can belong to, namely those along the path. This highlights an important difference between the CRP and the nCRP. That is that while the CRP is sufficient for drawing a community for an entity, the nCRP must be used alongside another stochastic process in order to determine the level along the path that the entity belongs to. This provides a segue to one such process, specifically the stick breaking process.

3.5 *The Stick Breaking Process*

The stick breaking process [62] is – like the CRP and nCRP – a Dirichlet process that draws its name from a metaphor which describes it. The metaphor starts by breaking a stick of unit length into two fragments at a point in the interval from 0 to 1 as drawn from the Beta distribution. One of the two fragments is preserved and the other fragment is broken again, analogously to the initial stick. This process is repeated an infinite number of times to yield an infinite number of fragments whose combined length is that of the initial stick. These fragments may be viewed as a probability distribution over the infinite sequence of discrete time-steps used to generate them. In other words, the stick breaking process is an infinite extension of the Dirichlet distribution insofar as while the Dirichlet distribution yields a probability distribution over L categories, the stick breaking process yields a probability distribution over an *infinite* number of categories. Formally, let the draw from the Beta distribution at the l^{th} iteration of the stick breaking process be denoted as $v^l \sim \text{Beta}(\mu\sigma, (1-\mu)\sigma)$. Thus, the lengths of the first fragment, denoted a^1, and its remainder are v^1 and $1 - v^1$, respectively. To obtain the length of the second fragment, a^2, draw v^1 and break off that fragment from what remains of the stick, namely $a^2 = v^1(1 - v^1)$. We define this process for an arbitrary l^{th}

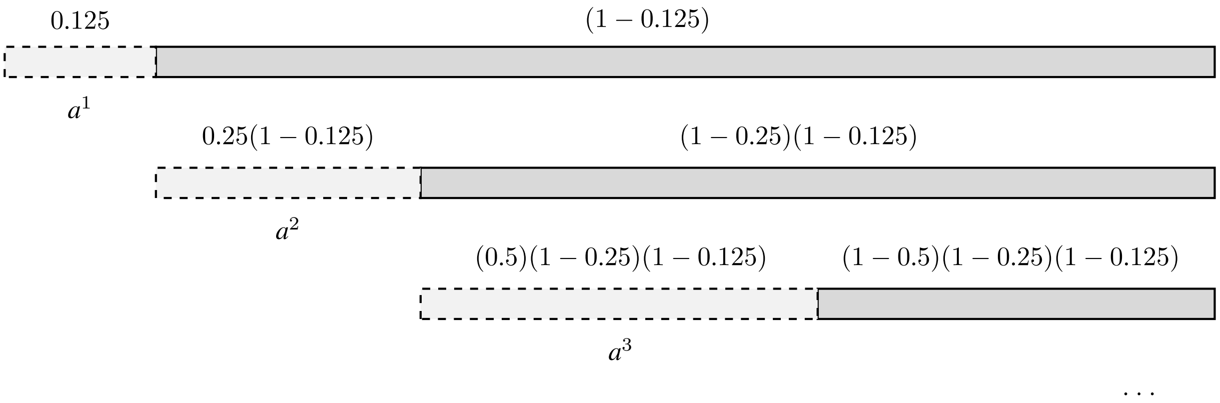


Fig. 4. Toy example of the stick breaking process with values $v^1 = 0.125$ $v^2 = 0.25$ $v^3 = 0.5$. Starting at the top of the figure, a unit length stick is broken at v^1. The remainder is then iteratively broken proportionally to draws from the Beta distribution.

time-step as follows:

$$a^l = v^l \prod_{k=1}^{l-1} (1 - v^k) \tag{7}$$

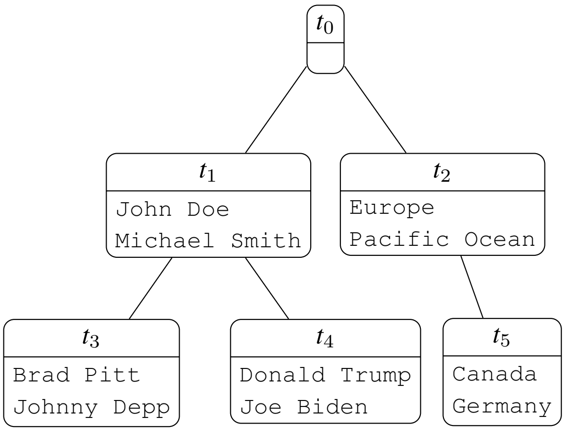
A concrete example involving the application of this rule is illustrated in Figure 4 which demonstrates the first three breaks of the stick along with the respective values of the broken fragments and their remainders. The realized stick fragments form a probability distribution in that $\sum_{l=1}^{\infty} a^l = 1$. We can thus define the probability mass function of the stick breaking process, denoted $\text{Stick}(\mu, \sigma)$, as follows:

$$\begin{aligned} \text{Stick}(\mu, \sigma) &= \sum_{l=1}^{\infty} a^l \\ &= \sum_{l=1}^{\infty} v^l \prod_{k=1}^{l-1} (1 - v^k) \end{aligned} \tag{8}$$

The stick breaking process is a generalization of the Griffiths-Engen-McCloskey distribution [61, 63] which may be seen as a special case where $\mu\sigma = 1$. The hyperparameters, $1 > \mu > 0$ and $\sigma > 0$, control the mean and variance of the distribution, respectively. Specifically, increasing μ values will pull the mean towards fragments broken later in the process and increasing σ values will increase the variance of the distribution. The resulting distribution can be used in conjunction with the nCRP to obtain a community for an entity given its sampled path. This is because by sampling the stick breaking distribution an index is obtained which can correspond to the level on the path that the entity belongs to. This motivates the use of the stick breaking process in our model. Namely, we use the stick breaking process as a prior over the levels in the induced hierarchy. We explain this in detail in the subsequent section.

4 Proposed Model

In describing our proposed model, we will adopt the notations used in the previous section to indicate the connection with the ideas discussed in the preliminaries. To aid in understanding, we first provide a summary of the components of our model before defining the generative process. This is followed by a formalization of the Gibbs sampling procedure and derivation of sampling equations.



Entity	**Path**	**Level**	**Community**
`Brad Pitt`	$t_1\ t_3$	2	t_3
`Canada`	$t_2\ t_5$	2	t_5
`Donald Trump`	$t_1\ t_4$	2	t_4
`Europe`	$t_2\ t_5$	1	t_2
`Germany`	$t_2\ t_5$	2	t_5
`Joe Biden`	$t_1\ t_4$	2	t_4
`John Doe`	$t_1\ t_4$	1	t_1
`Johnny Depp`	$t_1\ t_3$	2	t_3
`Michael Smith`	$t_1\ t_3$	1	t_1
`Pacific Ocean`	$t_2\ t_5$	1	t_2

Fig. 5. Toy example depicting a potential hierarchy induced by our model. The table on right side captures the path and level sampled for each entity in the knowledge graph as well as its corresponding community. The left side provides a visualization of this hierarchy.

4.1 Model Description

Like all stochastic blockmodels, our model is defined as a set of probability distributions such that when these distributions are sampled from, they generate the adjacency tensor of the knowledge graph. The choice of these distributions makes assumptions about the underlying structure that governs the graph's interactions. In devising our model, we assume a hierarchy of entity communities which are captured in the form of a tree. The entities in these communities interact with one another as a function of their membership to a community. In other words, interactions are modelled at the community level and extended downwards to their constituent entities. Unlike most stochastic blockmodels, these community relations are modelled with respect to a predicate in the knowledge graph. In other words, the interactions between entities are predicate dependant such that the degree of interaction between entities, changes depending on the predicate that links them. This allows the model to capture structures extending beyond those implied by mere interaction density. Thus, in order to generate the knowledge graph's adjacency tensor, we need to know its hierarchical community structure, its entities' memberships to communities, and the interactions between its communities. The induction of these components, which may be seen as a byproduct of the generative process, is the objective of our model. We note that the communities' constituent entities do not conform to is-a relationships as would be implied by the hierarchy. This is because the hierarchy is imposed on the communities themselves as opposed to their constituent entities. An example of this is highlighted in Figure 5 where the entity `Canada` is a descendant of the entity `Pacific Ocean`. Of course, `Canada` is not a `Pacific Ocean` however the concept modelled by community t_5, namely countries, is an instance of the concept modelled by community t_2, namely locations.

4.1.1 Community Memberships

Entities are assigned to communities through the conjunction of two variables: entity paths and level indicators. Paths define the tree structure over the community hierarchy by sampling from the nCRP as described in the previous section. We thus denote an entity path as $\mathbf{p}_i$ for entity e_i, such that $\mathbf{p}_i := [p_i^1, p_i^2, ..., p_i^L]$ where p_i^l represents the community at level l. We draw attention to the fact that this definition omits the root community from the path, namely p_i^0, since all entities must pass through it. It also allows a hierarchy with a depth of L to have entity path vectors of dimension L, simplifying the notation. Entity paths are drawn from the nCRP, denoted as $\mathbf{p}_i \sim \text{nCRP}(\gamma)$. Thus, all the entity paths sampled in the model form a $|\mathcal{E}| \times L$ matrix which we denote as $\mathbf{P}$. γ is the aforementioned hyperparameter of the nCRP and is responsible for controlling the probability of generating a new branch in the hierarchy as the path is being sampled. When a new branch is generated at level l such that $l > L$, $L - l$ new communities are also generated and populated solely by the sampling entity. Furthermore, if a path is resampled such that its corresponding entity obtains a new path which leaves behind empty communities, those empty communities

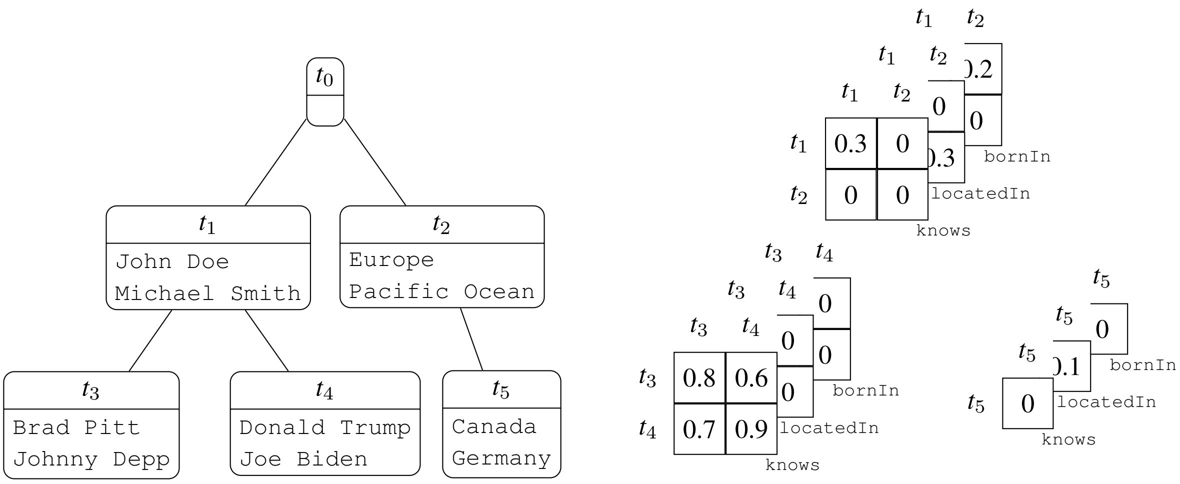


Fig. 6. The potential community relations induced by our model on the toy example introduced earlier. The hierarchy on the left of the figure has three sibling groups and three predicates: `knows`, `locatedIn`, and `bornIn`. The three tensors on the right correspond to the community relations of the three sibling groups.

are removed from the hierarchy. As such, the number of communities in the hierarchy is subject to constant change throughout the sampling process.

Having sampled entity paths, in order for entities to be assigned to communities, their levels must be obtained. Entity levels are modelled by two variables in our approach: level memberships and level indicators. Level memberships, denoted $\mathbf{a}_i$ for entity e_i, capture the probability of the entity's belonging to each of the L levels. As such, all the level memberships in our model form a $|\mathcal{E}| \times L$ matrix, $\mathbf{A}$. This is similar to the mixed-membership property of the Mixed Membership Stochastic Blockmodel wherein an entity has a membership distribution over all communities. The difference, as pointed out by Ho et al. [12], is that in hierarchical models this distribution is restricted to communities along the entity's sampled path, otherwise the process of obtaining paths, and indeed the hierarchy itself, would lose its meaning. Level memberships are drawn from the stick breaking process, $\mathbf{a}_i \sim \text{Stick}(\mu, \sigma)$ with hyperparameters μ and σ. Recall that this process yields an infinite distribution and must therefore be truncated to a dimension of L to correspond with the depth of the tree. The truncation is performed by removing all probabilities at levels greater than L and renormalizing. The distribution captured by an entity's level membership is used to sample its level indicator. The level indicator indicates the level to which an entity belongs and thus, in conjunction with its path, assigns it to a community. Level indicators are drawn in the context of an interaction between two entities. Specifically, when modelling the probability of an interaction from entity e_i to entity e_j we draw two level indicators, one for the sender entity and one for the receiver entity denoted as $z_{i\rightarrow j}$ and $z_{i\leftarrow j}$, respectively. The sender and receiver level indicators correspond to the levels of entities e_i and e_j in the context of their pairwise interaction. Thus, our model samples $|\mathcal{E}|^2$ sender and receiver level indicators each leading to two $|\mathcal{E}| \times |\mathcal{E}|$ matrices $\mathbf{Z}_{\rightarrow}$ and $\mathbf{Z}_{\leftarrow}$ for all the senders and receivers, respectively. To simplify notation in our inference procedure, we concatenate these matrices to form a $|\mathcal{E}| \times |\mathcal{E}| \times 2$ level indicator tensor, $\mathbf{Z}$. Since level memberships are themselves probability distributions, they may be sampled from directly to indicate an entity's level. Specifically, level indicators are drawn from multinomial distributions, namely $z_{i\rightarrow j} \sim \text{Multinomial}(\mathbf{a}_i)$ and $z_{i\leftarrow j} \sim \text{Multinomial}(\mathbf{a}_j)$, which yield one of the L levels in the hierarchy. The interplay between paths and levels when assigning entities to communities may be summarized as follows: paths identify a hierarchy of candidate communities and level indicators select one of the candidates for the entity. This dynamic is captured in the toy example in Figure 5.

4.1.2 Community Relations

Community relations describe the degree to which entities in any two communities are likely to interact with one another through a specific predicate. In other words, they model the probability of observing a value of one in the knowledge graph's adjacency tensor. These interactions are captured by a $\mathcal{T} \times \mathcal{T} \times \mathcal{R}$ tensor, denoted **C**, where $\mathcal{T}$ is the set of all communities in the hierarchy. We note that because the communities in $\mathcal{T}$ are a result of sampling from the nCRP and are thus subject to change with each successive sample, the dimensionality of **C** is also subject to change in the sampling process. This presents a challenge to our sampling scheme since it is possible to sample communities via the nCRP for which there are no community relation values. We overcome this issue through the marginalization of community relations as discussed in the subsequent subsection. The community relation c_{pqr} is an entry in **C** and captures the probability of interaction between entities in community t_p with entities in community t_q through predicate r_r. As such, the value of c_{pqr} is bounded to $1 \geqslant c_{pqr} \geqslant 0$. In order to preserve the hierarchical structure that was induced by sampling paths and levels, the community relations must be limited to take on non-zero values only when interacting with communities which are proximal to them in the hierarchy. This restriction is vital as allowing for interaction between any two communities in the hierarchy would render it meaningless and our model would be reduced to a fixed size mixed membership stochastic blockmodel such as the ones described in Section 2.

In restricting the values of community relations we take an approach similar to that of the Multiscale Community Blockmodel. Specifically, we borrow the concept of a sibling group which refers to a set of communities that share the same parent in the hierarchy. Only the community relations between communities in the same sibling group are modelled in our approach. Thus, when obtaining the interaction degree of two entities whose communities have the same parent, it is sufficient to merely access the corresponding value in **C**. When their communities do not share the same parent, a coarsening procedure is applied to obtain an interaction degree. The coarsening procedure traverses the paths of the two entities to find the deepest pair of communities which are in the same sibling group. Formally, to obtain the community relation degree from entity e_i to entity e_j on predicate r_r, we define the function $\Psi(i, j, r)$ as follows:

$$\Psi(i, j, r) = \begin{cases} c_{p_i^{z_{i\to j}} p_j^{z_{i\leftarrow j}} r} & p_i^{z_{i\to j}-1} = p_j^{z_{i\leftarrow j}-1} \\ c_{\Phi(i \mid j)\Phi(j \mid i) r} & p_i^{z_{i\to j}-1} \neq p_j^{z_{i\leftarrow j}-1} \end{cases} \tag{9}$$

Wherein $\Phi(i \mid j)$ and $\Phi(j \mid i)$ are functions that find the ancestor communities of entities e_i and e_j which share the same sibling group, respectively. These values are obtained by indexing the entities' paths on the level at which they diverge from their interaction partner. This process is made clear in their definitions:

$$\begin{aligned} \Phi(i \mid j) &= p_i^{min(\{l \,:\, p_i^l \neq p_j^l\})} \\ \Phi(j \mid i) &= p_j^{min(\{l \,:\, p_i^l \neq p_j^l\})} \end{aligned} \tag{10}$$

This approach differs from the Multiscale Community Blockmodel in that while there are restricted entries in **C**, these values are never accessed. Instead, all communities are coarsened to an ancestor which allows for their interaction to take on a non-restricted value.

Community relations are drawn from the Beta distribution parameterized by $\lambda > 0$ and $\eta > 0$, and denoted as $c_{pqr} \sim \text{Beta}(\lambda, \eta)$. This ensures that community relations take on probability values which can be used in conjunction with the Bernoulli distribution. λ and η are hyperparameters of our model and determine the density of the generated knowledge graph such that increasing λ values with respect to η yields denser results. Figure 6 provides a visualization of a potential sampling of community relations. We note that the three tensors correspond to the three sibling groups for which community relations take on non-restricted values. The diagonal and off-diagonal values in these tensors represent the intra and inter community relations, respectively. Thus, based on these values, there is a probability of 0.8 that `Brad Pitt knows Johnny Depp` and a probability of 0.6 that `Brad Pitt knows Donald Trump`. We provide an exploration of the recovered community relations on real-world data in Section 5.

4.1.3 Generative Process

The generative process of our model refers to the sequential sampling of components which allow for the generation of the target knowledge graph. In other words, the goal is to draw a binary value for each $g_{ijr} \in \mathbf{G}$ such that it

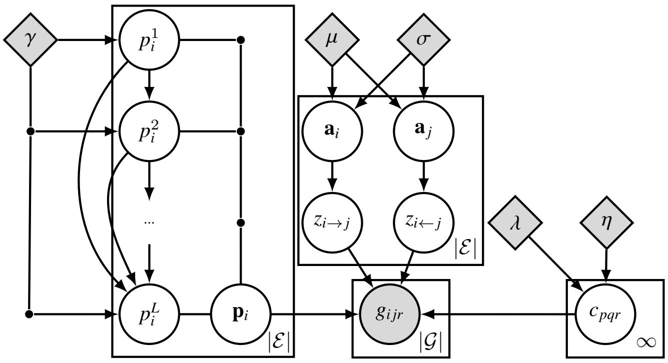


Fig. 7. Plate diagram for our model. Circles indicate random variables whereas diamonds indicate model hyperparameters. Shading indicates observed variables and lack of shading indicates latent variables.

equals the knowledge graph's adjacency tensor. But before this can be done, it is necessary to sample the variables it is dependent on. The first components sampled in the generative process are the paths and level memberships for each entity in the knowledge graph from the nCRP and stick distributions, respectively. Having drawn the paths, we now have the set of communities in the hierarchy and can draw community relations from the Beta distribution. At this point in the generative process, entities are not yet assigned to communities. The community memberships for these entities have been drawn, however, allowing for the sampling of community levels for each pair of entities in the knowledge graph from the multinomial distribution. With the community levels drawn, all the components for generating the knowledge graph are in place. The binary value for the interaction from entity e_i to entity e_j on predicate r_r is drawn from the Bernoulli distribution using each entity's respective community's interactions, namely $g_{ijr} \sim \text{Bernoulli}(\Psi(i, j, r))$. The plate diagram for this process is illustrated in Figure 7 and the formal definition is as follows:

- For each entity in the knowledge graph; $e_i \in \mathcal{E}$
    - $\mathbf{p}_i \sim \text{nCRP}(\gamma)$
    - $\mathbf{a}_i \sim \text{Stick}(\mu, \sigma)$
- For each sender community in the hierarchy; $t_p \in \mathcal{T}$
    - For each receiver community in the hierarchy; $t_q \in \mathcal{T}$
        - For each predicate in the knowledge graph; $r_r \in \mathcal{R}$
            - $c_{pqr} \sim \text{Beta}(\lambda, \eta)$
- For each sender entity in the knowledge graph; $e_i \in \mathcal{E}$
    - For each receiver entity in the knowledge graph; $e_j \in \mathcal{E}$
        - $z_{i\rightarrow j} \sim \text{Multinomial}(\mathbf{a}_i)$
        - $z_{i\leftarrow j} \sim \text{Multinomial}(\mathbf{a}_j)$
        - For each predicate in the knowledge graph; $r_r \in \mathcal{R}$
            - $g_{ijr} \sim \text{Bernoulli}(\Psi(i, j, r))$

We note that this process is unsupervised and does not impose any assumptions about the partition of entities to communities or the structure of the hierarchy other than to limit its depth. In fact, the depth is the only constraint imposed on the generative process. The other hyperparameters which must be specified a priori – namely γ, μ, σ, λ, and η – merely influence the prior distributions of our model. They may pull the latent variables in the assumed direction but only insofar as the data allows it. This, recall, is due to the sampling of latent variables from their posterior distribution which is conditioned on the data. As a result, with a strong enough likelihood, the effects of the hyperparameters and the prior relatively diminish. As with most stochastic blockmodels, the exact inference for our model is intractable and must be approximated using an inference scheme. For this we adopt collapsed Gibbs sampling, an extension of the aforementioned Gibbs sampling.

4.2 Collapsed Gibbs Sampling

Collapsed Gibbs sampling refers to an extension of Gibbs sampling in which a subset of model variables are marginalized over and therefore do not need to be sampled directly. These variables are said to be collapsed out of the Gibbs sampler. Collapsing of these variables is done analytically via integration and ensures a faster mixing process. This is because the calculation of probability distributions for sampling is generally computationally expensive. Having fewer variables then leads to a faster arrival at the desired stationary distribution. Furthermore, the calculation of probability distributions which have not been collapsed out of the sampling process is generally faster in collapsed Gibbs sampling. This is because in regular Gibbs sampling draws are made from the full conditionals of variables. In collapsed Gibbs sampling, collapsed variables have been integrated out of the process and the remaining variables are conditioned on a lower-dimensional space. Collapsing of variables is usually tractable when they are the conjugate prior of their dependent variables. In our model, community relations and level memberships are both conjugate priors of their dependant variables, namely level indicators and entity relations, respectively. We leverage these conjugacies to marginalize over these two variables in our sampling process. After marginalization, the sampling equations may be derived for the remaining variables.

4.2.1 Marginalizing Community Relations

In order to marginalize out community relations, it is necessary to find a closed form solution which allows for integration during path sampling. To this end, we can leverage the Bernoulli-Beta conjugacy which ensures that given a Bernoulli likelihood and Beta prior, the posterior will also be drawn from the Beta distribution. Employing this conjugacy is possible due to the formulation of our model in which entity relations are drawn from the Bernoulli distribution and community relations assume a Beta prior. We see this explicitly when applying Bayes' theorem to obtain the posterior as follows:

$$\mathbb{P}(c_{pqr} \mid \mathbf{C}_{-(pqr)}, \mathbf{G}, \mathbf{P}, \mathbf{Z}, \lambda, \eta) = \frac{\mathbb{P}(\mathbf{G} \mid \mathbf{C}, \mathbf{P}, \mathbf{Z}, \lambda, \eta)\mathbb{P}(c_{pqr} \mid \mathbf{C}_{-(pqr)}, \lambda, \eta)}{\int_{c_{pqr}} \mathbb{P}(\mathbf{G} \mid \mathbf{C}, \mathbf{P}, \mathbf{Z}, \lambda, \eta)\mathbb{P}(c_{pqr} \mid \mathbf{C}_{-(pqr)}, \lambda, \eta) \, \mathrm{d}c_{pqr}} \tag{11}$$

Where $\mathbb{P}(\mathbf{G} \mid \mathbf{C}, \mathbf{P}, \mathbf{Z}, \lambda, \eta)$ is the likelihood of generating entity relations and $\mathbb{P}(c_{pqr} \mid \mathbf{C}_{-(pqr)}, \lambda, \eta)$ is the prior placed on community relations. $\mathbf{C}_{-(pqr)}$ indicates the community relations tensor $\mathbf{C}$ without c_{pqr}. Before proceeding we introduce helper variables $\#^{c_{pqr}=1}$ and $\#^{c_{pqr}=0}$ to indicate the number of existing and non-existing interactions between entities from community t_p to community t_q on predicate r_r, respectively:

$$\begin{aligned} \#^{c_{pqr}=1} &= \left| \left\{ g_{xyz} \in \mathbf{G} \; : \; \Psi(x, y, z) = c_{pqr} \wedge g_{xyz} = 1 \right\} \right| \\ \#^{c_{pqr}=0} &= \left| \left\{ g_{xyz} \in \mathbf{G} \; : \; \Psi(x, y, z) = c_{pqr} \wedge g_{xyz} = 0 \right\} \right| \end{aligned} \tag{12}$$

We can now derive a closed-form solution for the posterior of community relations by applying the distributions defined in our model:

$$\begin{aligned} \mathbb{P}(c_{pqr} \mid \mathbf{C}_{-(pqr)}, \mathbf{G}, \mathbf{P}, \mathbf{Z}, \lambda, \eta) &\overset{(1)}{=} \frac{\left(\prod_{g_{xyz} \in \mathbf{G}} \text{Bernoulli}(c_{pqr}, 1 - c_{pqr})\right)\left(\text{Beta}(\lambda, \eta)\right)}{\int_{c_{pqr}} \left(\prod_{g_{xyz} \in \mathbf{G}} \text{Bernoulli}(c_{pqr}, 1 - c_{pqr})\right)\left(\text{Beta}(\lambda, \eta)\right) \mathrm{d}c_{pqr}} \\ &\overset{(2)}{=} \frac{\left(c_{pqr}^{\#^{c_{pqr}=1}} (1 - c_{pqr})^{\#^{c_{pqr}=0}}\right)\left(\dfrac{c_{pqr}^{\lambda-1}(1 - c_{pqr})^{\eta-1}}{\mathrm{B}(\lambda, \eta)}\right)}{\int_{c_{pqr}} \left(c_{pqr}^{\#^{c_{pqr}=1}} (1 - c_{pqr})^{\#^{c_{pqr}=0}}\right)\left(\dfrac{c_{pqr}^{\lambda-1}(1 - c_{pqr})^{\eta-1}}{\mathrm{B}(\lambda, \eta)}\right) \mathrm{d}c_{pqr}} \end{aligned}$$

$$\overset{(3)}{=} \frac{c_{pqr}^{\#^{c_{pqr}=1}+\lambda-1}(1-c_{pqr})^{\#^{c_{pqr}=0}+\eta-1}}{\int_{c_{pqr}} c_{pqr}^{\#^{c_{pqr}=1}+\lambda-1}(1-c_{pqr})^{\#^{c_{pqr}=0}+\eta-1}\,\mathrm{d}c_{pqr}}$$

$$\overset{(4)}{=} \frac{c_{pqr}^{\#^{c_{pqr}=1}+\lambda-1}(1-c_{pqr})^{\#^{c_{pqr}=0}+\eta-1}}{\mathrm{B}(\#^{c_{pqr}=1}+\lambda, \#^{c_{pqr}=0}+\eta)}$$

$$= \mathrm{Beta}(\#^{c_{pqr}=1}+\lambda, \#^{c_{pqr}=0}+\eta) \tag{13}$$

Such that in the derivation above: (1) is the Bayes' theorem definition as per Equation 11 using the probability distributions defined in our model; (2) uses the probability masses and densities of the Bernoulli and Beta distributions as per Equations A.1 and A.3; (3) is obtained by applying power rules and dividing out the Beta function which is constant with respect to c_{pqr} in the integral; and (4) utilizes the integral form of the Beta function as derived in Equation B.1. The posterior as defined in Equation 13 allows for community relations to be integrated out when sampling paths. As such they are not sampled directly in the inference process.

4.2.2 Marginalizing Level Memberships

There are two ways in which to approach marginalizing level memberships in our model. Firstly, Sethuraman [62] showed that the realization of the stick breaking process follows the Dirichlet distribution. We can leverage this because, in practice, the dimensionality of the level memberships gets bounded to the depth of the tree, L. It is therefore possible to model level memberships with an L dimensional Dirichlet distribution. As discussed in Ho et al. [12], this prior has the disadvantage of either being too expressive or not expressive enough depending on its parameterization. Regardless, we show the marginalization of this case in Appendix D. In this subsection, however, we focus on the infinite case using the stick breaking process as defined in our model. To this end, we use the multinomial-stick conjugacy to obtain a stick breaking posterior which is used as the prior for the level indicators later on. The posterior is defined as follows:

$$\begin{aligned}\mathbb{P}(\mathbf{a}_i \mid \mathbf{A}_{-i}, \mathbf{Z}, \mu, \sigma) &= \frac{\mathbb{P}(\mathbf{Z} \mid \mathbf{A}, \mu, \sigma)\mathbb{P}(\mathbf{a}_i \mid \mathbf{A}_{-i}, \mu, \sigma)}{\int_{\mathbf{a}_i} \mathbb{P}(\mathbf{Z} \mid \mathbf{A}, \mu, \sigma)\mathbb{P}(\mathbf{a}_i \mid \mathbf{A}_{-i}, \mu, \sigma)\,\mathrm{d}\mathbf{a}_i} \\ &= \frac{\text{Multinomial}(\mathbf{a}_i)\text{Stick}(\mu, \sigma)}{\int_{\mathbf{a}_i} \text{Multinomial}(\mathbf{a}_i)\text{Stick}(\mu, \sigma)\,\mathrm{d}\mathbf{a}_i}\end{aligned} \tag{14}$$

Where $\mathbb{P}(\mathbf{Z} \mid \mathbf{A}, \mu, \sigma)$ and $\mathbb{P}(\mathbf{a}_i \mid \mathbf{A}_{-i}, \mu, \sigma)$ are the likelihood and prior of level memberships, respectively. We use the definitions from our model and replace these with the multinomial and stick breaking distributions. Before proceeding, we define $\mathbf{z}_{i*} = \{z_{x\leftrightarrow y} \in \mathbf{Z} : x = i \vee y = i\}$ representing all level indicators for entity e_i. In this notation $z_{i\leftrightarrow j} := z_{i\to j} \vee z_{i\leftarrow j}$ is used as shorthand for any level indicator relating entity e_i with entity e_j regardless of which entities are taking on the sender and receiver roles. This allows for defining two helper variables, $\#^{\mathbf{z}_{i*}=l}$ and $\#^{\mathbf{z}_{i*}>l}$, to indicate the number of indicators in $\mathbf{z}_{i*}$ at and below level l in the hierarchy, respectively:

$$\begin{aligned}\#^{\mathbf{z}_{i*}=l} &= \left|\left\{z_{i\leftrightarrow j} \in \mathbf{z}_{i*} : z_{i\leftrightarrow j} = l\right\}\right| \\ \#^{\mathbf{z}_{i*}>l} &= \left|\left\{z_{i\leftrightarrow j} \in \mathbf{z}_{i*} : z_{i\leftrightarrow j} > l\right\}\right|\end{aligned} \tag{15}$$

With these definitions in place, we can derive the stick breaking posterior of level memberships:

$$\mathbb{P}(\mathbf{a}_i \mid \mathbf{A}_{-i}, \mathbf{Z}, \mu, \sigma)$$

$$\overset{(1)}{=} \frac{\left(\frac{\Gamma(\sum_{l=1}^{\infty} \#^{\mathbf{z}_{i*}=l}+1))}{\prod_{l=1}^{\infty}\Gamma(\#^{\mathbf{z}_{i*}=l}+1))}\prod_{l=1}^{\infty} a_i^l\right)\left(\sum_{l=1}^{\infty} v_l \prod_{k=1}^{l-1}(1-v_k)\mathbb{I}(a_i^l=l)\right)}{\int_{\mathbf{V}}\left(\frac{\Gamma(\sum_{l=1}^{\infty} \#^{\mathbf{z}_{i*}=l}+1))}{\prod_{l=1}^{\infty}\Gamma(\#^{\mathbf{z}_{i*}=l}+1))}\prod_{l=1}^{\infty} a_i^l\right)\left(\sum_{l=1}^{\infty} v_l \prod_{k=1}^{l-1}(1-v_k)\mathbb{I}(a_i^l=l)\right)\mathrm{d}^{\infty}\mathbf{V}}$$

$$\overset{(2)}{=} \frac{\left(\frac{\Gamma(\sum_{l=1}^{\infty} \#^{\mathbf{z}_{i*}=l}+1))}{\prod_{l=1}^{\infty}\Gamma(\#^{\mathbf{z}_{i*}=l}+1))}\prod_{l=1}^{\infty}\left(v_l\prod_{k=1}^{l-1}(1-v_k)\right)^{\#^{\mathbf{z}_{i*}=l}}\right)\left(\mathrm{Beta}(\mu\sigma,(1-\mu)\sigma)\prod_{k=1}^{l-1}\mathrm{Beta}((1-\mu)\sigma,\mu\sigma)\right)}{\int_{\mathbf{V}}\left(\frac{\Gamma(\sum_{l=1}^{\infty} \#^{\mathbf{z}_{i*}=l}+1))}{\prod_{l=1}^{\infty}\Gamma(\#^{\mathbf{z}_{i*}=l}+1))}\prod_{l=1}^{\infty}\left(v_l\prod_{k=1}^{l-1}(1-v_k)\right)^{\#^{\mathbf{z}_{i*}=l}}\right)\left(\mathrm{Beta}(\mu\sigma,(1-\mu)\sigma)\prod_{k=1}^{l-1}\mathrm{Beta}((1-\mu)\sigma,\mu\sigma)\right)\mathrm{d}^{\infty}\mathbf{V}}$$

$$\overset{(3)}{=} \frac{\left(\frac{\Gamma(\sum_{l=1}^{\infty} \#^{\mathbf{z}_{i*}=l}+1))}{\prod_{l=1}^{\infty}\Gamma(\#^{\mathbf{z}_{i*}=l}+1))}v_l^{\#^{\mathbf{z}_{i*}=l}}(1-v_l)^{\#^{\mathbf{z}_{i*}>l}}\prod_{k=1}^{l-1}v_k^{\#^{\mathbf{z}_{i*}>k}}(1-v_k)^{\#^{\mathbf{z}_{i*}=k}}\right)}{\int_{\mathbf{V}}\left(\frac{\Gamma(\sum_{l=1}^{\infty} \#^{\mathbf{z}_{i*}=l}+1))}{\prod_{l=1}^{\infty}\Gamma(\#^{\mathbf{z}_{i*}=l}+1))}v_l^{\#^{\mathbf{z}_{i*}=l}}(1-v_l)^{\#^{\mathbf{z}_{i*}>l}}\prod_{k=1}^{l-1}v_k^{\#^{\mathbf{z}_{i*}>k}}(1-v_k)^{\#^{\mathbf{z}_{i*}=k}}\right)}$$

$$\frac{\left(\frac{v_l^{\mu\sigma-1}(1-v_l)^{(1-\mu)\sigma-1}}{\mathrm{B}(\mu\sigma,(1-\mu)\sigma)}\prod_{k=1}^{l-1}\frac{v_k^{(1-\mu)\sigma-1}(1-v_k)^{\mu\sigma-1}}{\mathrm{B}((1-\mu)\sigma,\mu\sigma)}\right)}{\left(\frac{v_l^{\mu\sigma-1}(1-v_l)^{(1-\mu)\sigma-1}}{\mathrm{B}(\mu\sigma,(1-\mu)\sigma)}\prod_{k=1}^{l-1}\frac{v_k^{(1-\mu)\sigma-1}(1-v_k)^{\mu\sigma-1}}{\mathrm{B}((1-\mu)\sigma,\mu\sigma)}\right)\mathrm{d}^{\infty}\mathbf{V}}$$

$$\overset{(4)}{=} \frac{v_l^{\#^{\mathbf{z}_{i*}=l}+\mu\sigma-1}(1-v_l)^{\#^{\mathbf{z}_{i*}>l}+(1-\mu)\sigma-1}\prod_{k=1}^{l-1}v_k^{\#^{\mathbf{z}_{i*}>k}+(1-\mu)\sigma-1}(1-v_k)^{\#^{\mathbf{z}_{k*}=k}+\mu\sigma-1}}{\int_{\mathbf{V}}v_l^{\#^{\mathbf{z}_{i*}=l}+\mu\sigma-1}(1-v_l)^{\#^{\mathbf{z}_{i*}>l}+(1-\mu)\sigma-1}\prod_{k=1}^{l-1}v_k^{\#^{\mathbf{z}_{i*}>k}+(1-\mu)\sigma-1}(1-v_k)^{\#^{\mathbf{z}_{k*}=k}+\mu\sigma-1}\mathrm{d}^{\infty}\mathbf{V}}$$

$$\overset{(5)}{=} \frac{v_l^{\#^{\mathbf{z}_{i*}=l}+\mu\sigma-1}(1-v_l)^{\#^{\mathbf{z}_{i*}>l}+(1-\mu)\sigma-1}\prod_{k=1}^{l-1}v_k^{\#^{\mathbf{z}_{i*}>k}+(1-\mu)\sigma-1}(1-v_k)^{\#^{\mathbf{z}_{k*}=k}+\mu\sigma-1}}{\mathrm{B}(\#^{\mathbf{z}_{i*}=l}+\mu\sigma,\#^{\mathbf{z}_{i*}>l}+(1-\mu)\sigma)\prod_{k=1}^{l-1}\mathrm{B}(\#^{\mathbf{z}_{i*}=k}+(1-\mu)\sigma,\#^{\mathbf{z}_{k*}=k}+\mu\sigma-1)}$$

$$= \mathrm{Beta}(\#^{\mathbf{z}_{i*}=l}+\mu\sigma,\#^{\mathbf{z}_{i*}>l}+(1-\mu)\sigma)\prod_{k=1}^{l-1}\mathrm{Beta}(\#^{\mathbf{z}_{i*}>k}+(1-\mu)\sigma,\#^{\mathbf{z}_{i*}=k}+\mu\sigma)$$

$$= \sum_{l=1}^{\infty} v_l \prod_{k=1}^{l-1}(1-v_k) \mid \mathbf{z}_{i*}$$

$$= \text{Stick}(\mu\sigma, (1-\mu)\sigma) \mid \mathbf{z}_{i*} \tag{16}$$

Where (1) is an application of the definitions of the Multinomial and stick breaking distributions as per Equations A.2 and 8 to the posterior as per Equation 14. (2) redefines the likelihood in terms of the Beta samples of the stick breaking process and the prior leverages the mirror symmetry property of the Beta function. The summation and indicator function are both removed at this stage to enhance readability. (3) rearranges the likelihood and substitutes the probability density function of the Beta distribution as per Equation A.3. (4) involves cancelling out terms in the numerator and denominator which are constant with respect to the integration. (5) utilizes the integral form of the Beta function as per Equation B.1 and the remainder of the derivation merely reverses the definitions applied earlier to arrive at the definition of the stick breaking process. We use the notation $\mid \mathbf{z}_{i*}$ to denote that the distribution preceding the notation is conditioned on $\mathbf{z}_{i*}$.

4.2.3 Sampling Entity Paths

Entity paths are one of the two variables which remain after collapsing the Gibbs sampler and must therefore be sampled directly. To sample a path for entity e_i, it must first be removed from the hierarchy, thereby allowing for its full conditional distribution to be obtained. The set of paths after having removed path $\mathbf{p}_i$ is denoted as $\mathbf{P}_{-i}$. We derive the posterior distribution of $\mathbf{p}_i$ by applying Bayes' theorem:

$$\begin{aligned}\mathbb{P}(\mathbf{p}_i \mid \mathbf{P}_{-i}, \mathbf{G}, \mathbf{Z}, \gamma, \lambda, \eta) &= \frac{\mathbb{P}(\mathbf{G}_{i*} \mid \mathbf{G}_{-(i*)}, \mathbf{P}, \mathbf{Z}, \gamma, \lambda, \eta)\mathbb{P}(\mathbf{p}_i \mid \mathbf{P}_{-i}, \gamma)}{\int_{\mathbf{p}_i} \mathbb{P}(\mathbf{G}_{i*} \mid \mathbf{G}_{-(i*)}, \mathbf{P}, \mathbf{Z}, \gamma, \lambda, \eta)\mathbb{P}(\mathbf{p}_i \mid \mathbf{P}_{-i}, \gamma)\, \mathrm{d}\mathbf{p}_i} \\ &\propto \mathbb{P}(\mathbf{G}_{i*} \mid \mathbf{G}_{-(i*)}, \mathbf{P}, \mathbf{Z}, \gamma, \lambda, \eta)\mathbb{P}(\mathbf{p}_i \mid \mathbf{P}_{-i}, \gamma)\end{aligned} \tag{17}$$

Where $\mathbf{G}_{i*} = \{g_{xyz} \in \mathbf{G} : i = x \ \vee \ i = y\}$ denotes all the triples in the knowledge graph that depend on path $\mathbf{p}_i$ and $\mathbf{G}_{-(i*)} = \mathbf{G} \setminus \mathbf{G}_{i*}$ is its complement. The integral form of the marginal distribution for generating the data is a normalizing constant for the posterior distribution. Calculating this integral is not necessary and we can instead sample paths from its proportional distribution as per Equation 17. The prior for sampling an entity path, $\mathbb{P}(\mathbf{p}_i \mid \mathbf{P}_{-i}, \gamma)$, is obtained from the nCRP. We note that due to the iterative nature of the Gibbs sampler, a path for entity e_i may already exist in the hierarchy from a previous iteration. As such, it must first be removed, hence the conditioning on $\mathbf{P}_{-i}$. We use $\mathbf{p}_i = t_q$ as a shorthand to indicate the path that terminates at community t_q, in other words $\mathbf{p}_i = t_q$ if $p_i^L = t_q$. Thus, the prior is calculated as follows:

$$\begin{aligned}\mathbb{P}(\mathbf{p}_i = t_q \mid \mathbf{P}_{-i}, \gamma) &= \mathbb{P}(p_i^L = t_q \mid \mathbf{P}_{-i}, \gamma) \\ &= \mathbb{E}\Big[(\mathbb{I}(p_i^L = t_q) \,\Big|\, \mathbf{P}_{-i}, \gamma)\Big] \\ &= \mathbb{P}(p_i^L = t_q \mid \mathbf{p}_i^{1\,:\,L-1}, \mathbf{P}_{-i}, \gamma) \prod_{l=1}^{L-1} \mathbb{P}(p_i^l = t_q^l \mid \mathbf{p}_i^{1\,:\,l-1}, \mathbf{P}_{-i}, \gamma)\end{aligned} \tag{18}$$

Where t_q^l is the ancestor community of community t_q at level l. Equation 18 requires the distribution for sampling a community conditioned on a partially sampled path. Recall that this is defined by the nCRP in Equation 6 and can be adapted here. Specifically, we calculate the probability of taking community t_q on level l having already sampled its path up to level $l-1$ as:

$$\mathbb{P}(p_i^l = t_q \mid \mathbf{p}_i^{1\,:\,l-1}, \mathbf{P}_{-i}, \gamma) = \begin{cases} \dfrac{\#_{-i}^{t_q}}{\#_{-i}^{t_q^{l-1}} + \gamma} & t_q \in \mathcal{T}_{-i} \\[2ex] \dfrac{\gamma}{\#_{-i}^{t_q^{l-1}} + \gamma} & t_q \notin \mathcal{T}_{-i} \end{cases} \tag{19}$$

Where $\#^{t_q}_{-i}$ extends the notation defined earlier to indicate the number of entities that have gone through community t_q in the hierarchy with path $\mathbf{p}_i$ removed. $\mathcal{T}_{-i}$ indicates all the communities in the hierarchy after path $\mathbf{p}_i$ has been removed. We note that this process requires the sampling of a path to start at the root community and proceed sequentially to a leaf community. Having obtained the prior, it is necessary to update the belief about the posterior with the data via the likelihood. The likelihood given a sampled path, $\mathbb{P}(\mathbf{G}_{i*} \mid \mathbf{G}_{-(i*)}, \mathbf{P}, \mathbf{Z}, \gamma, \lambda, \eta)$, is defined with the help of the following helper variables:

$$\begin{aligned}
\mathbf{C}_{i*} &= \Big\{c_{pqr} \in \mathbf{C} \ : \ (\exists g_{xyz} \in \mathbf{G}_{i*} : \Psi(x,y,z) = c_{pqr})\Big\} \\
\#^{c_{pqr}=1}_{-i} &= \Big|\Big\{g_{xyz} \in \mathbf{G}_{-(i*)} : \Psi(x,y,z) = c_{pqr} \wedge g_{xyz} = 1\Big\}\Big| \\
\#^{c_{pqr}=0}_{-i} &= \Big|\Big\{g_{xyz} \in \mathbf{G}_{-(i*)} : \Psi(x,y,z) = c_{pqr} \wedge g_{xyz} = 0\Big\}\Big| \\
\#^{c_{pqr}=1}_{i} &= \Big|\Big\{g_{xyz} \in \mathbf{G}_{i*} : \Psi(x,y,z) = c_{pqr} \wedge g_{xyz} = 1\Big\}\Big| \\
\#^{c_{pqr}=0}_{i} &= \Big|\Big\{g_{xyz} \in \mathbf{G}_{i*} : \Psi(x,y,z) = c_{pqr} \wedge g_{xyz} = 0\Big\}\Big|
\end{aligned} \tag{20}$$

The definitions above capture the following: $\mathbf{C}_{i*}$ is the set of communities dependant on an interaction in $\mathbf{G}_{i*}$; $\#^{c_{pqr}=1}_{-i}$ and $\#^{c_{pqr}=1}_{i}$ are the counts of existing entity relations from communities t_p to t_q in $\mathbf{G}_{-(i)}$ and $\mathbf{G}_i$, respectively; and $\#^{c_{pqr}=0}_{-i}$ and $\#^{c_{pqr}=0}_{i}$ are the counts of non-existing entity relations from communities t_p to t_q in $\mathbf{G}_{-(i)}$ and $\mathbf{G}_i$, respectively. In the discrete space, the likelihood is understood as the joint probability of generating the data as per a probability mass function. In our model, the data is obtained by drawing from the Bernoulli distribution conditioned on community relations. Recall that these parameters are marginalized out of our model and thus never sampled directly. As such, in order to calculate the likelihood we must integrate with respect to the community relations. This is possible by leveraging Equation 13 which allows us to obtain a closed form solution:

$$\begin{aligned}
&\mathbb{P}(\mathbf{G}_{i*} \mid \mathbf{G}_{-(i*)}, \mathbf{P}, \mathbf{Z}, \gamma, \lambda, \eta) \\
&\quad = \prod_{c_{pqr} \in \mathbf{C}_{i*}} \int_{c_{pqr}} \mathbb{P}(\mathbf{G}_{i*} \mid \mathbf{G}_{-(i*)}, \mathbf{C}, \mathbf{P}, \mathbf{Z}, \gamma, \mu, \sigma, \lambda, \eta) \mathbb{P}(c_{pqr} \mid \mathbf{C}_{-(pqr)}, \mathbf{G}_{-(i*)}, \mathbf{P}, \mathbf{Z}, \lambda, \eta) \, \mathrm{d}c_{pqr} \\
&\quad \overset{(1)}{=} \prod_{c_{pqr} \in \mathbf{C}_{i*}} \int_{c_{pqr}} \left(\prod_{g_{xyz} \in \mathbf{G}_{i*}} \text{Bernoulli}(c_{pqr}, 1 - c_{pqr}) \right) \left(\text{Beta}(\#^{c_{pqr}=1}_{-i} + \lambda, \#^{c_{pqr}=0}_{-i} + \eta) \right) \mathrm{d}c_{pqr} \\
&\quad \overset{(2)}{=} \prod_{c_{pqr} \in \mathbf{C}_{i*}} \int_{c_{pqr}} \left(c_{pqr}^{\#^{c_{pqr}=1}_{i}} (1 - c_{pqr})^{\#^{c_{pqr}=0}_{i}} \right) \left(\frac{c_{pqr}^{\#^{c_{pqr}=1}_{-i} + \lambda - 1} (1 - c_{pqr})^{\#^{c_{pqr}=0}_{-i} + \eta - 1}}{\mathrm{B}(\#^{c_{pqr}=1}_{-i} + \lambda, \#^{c_{pqr}=0}_{-i} + \eta)} \right) \mathrm{d}c_{pqr} \\
&\quad = \prod_{c_{pqr} \in \mathbf{C}_{i*}} \frac{1}{\mathrm{B}(\#^{c_{pqr}=1}_{-i} + \lambda, \#^{c_{pqr}=0}_{-i} + \eta)} \int_{c_{pqr}} c_{pqr}^{\#^{c_{pqr}=1}_{i} + \#^{c_{pqr}=1}_{-i} + \lambda - 1} (1 - c_{pqr})^{\#^{c_{pqr}=0}_{i} + \#^{c_{pqr}=0}_{-i} + \eta - 1} \, \mathrm{d}c_{pqr} \\
&\quad \overset{(3)}{=} \prod_{c_{pqr} \in \mathbf{C}_{i*}} \frac{\mathrm{B}(\#^{c_{pqr}=1}_{i} + \#^{c_{pqr}=1}_{-i} + \lambda, \#^{c_{pqr}=0}_{i} + \#^{c_{pqr}=0}_{-i} + \eta)}{\mathrm{B}(\#^{c_{pqr}=1}_{-i} + \lambda, \#^{c_{pqr}=0}_{-i} + \eta)} \\
&\quad \overset{(4)}{=} \prod_{c_{pqr} \in \mathbf{C}_{i*}} \left(\frac{\Gamma(\#^{c_{pqr}=1}_{i} + \#^{c_{pqr}=1}_{-i} + \lambda) \Gamma(\#^{c_{pqr}=0}_{i} + \#^{c_{pqr}=0}_{-i} + \eta)}{\Gamma(\#^{c_{pqr}=1}_{i} + \#^{c_{pqr}=1}_{-i} + \#^{c_{pqr}=0}_{i} + \#^{c_{pqr}=0}_{-i} + \lambda + \eta)} \right) \left(\frac{\Gamma(\#^{c_{pqr}=1}_{-i} + \#^{c_{pqr}=0}_{-i} + \lambda + \eta)}{\Gamma(\#^{c_{pqr}=1}_{-i} + \lambda) \Gamma(\#^{c_{pqr}=0}_{-i} + \eta)} \right)
\end{aligned} \tag{21}$$

In the derivation above: (1) the prior probability of drawing c_{pqr} is obtained from Equation 13; (2) utilizes the definitions as per Equations A.1 and A.3 as well as the helper variables introduced in Equation 20; (3) leverages the integral form of the Beta function as per Equation B.1; and (4) expands the Beta function to its Gamma formulation as per Equation A.3. Having derived the prior and likelihood in Equations 18 and 21, respectively, it is possible to sample from Equation 17 to obtain entity paths in our model. The time complexity of sampling one such path is $\mathcal{O}(|\mathcal{E}|^2|\mathcal{R}|L)$. This is due to the fact that it is necessary to obtain a sampling probability for all potential paths in the hierarchy, which has a bound of $|\mathcal{E}|L$ in the case where each entity takes a unique path. For each of these potential paths, the iteration through all $|\mathcal{E}|$ entities and $|\mathcal{R}|$ predicates is required to determine the effect on the likelihood selecting such a path would have.

4.2.4 Sampling Level Indicators

Level indicators are drawn from the multinomial distribution conditioned on level memberships. Recall that level memberships were marginalized over in our inference scheme using the multinomial-stick conjugacy and are thus never sampled directly. Nevertheless, we draw them indirectly when computing the prior for level indicators. As with sampling paths, we obtain the distribution proportional to that of level indicators by Bayes' rule. In what follows, we provide the derivation for the posterior of $z_{i\to j}$ and note that given its structural symmetry, $z_{i\leftarrow j}$ is derived analogously. The posterior distribution of $z_{i\to j}$ is expressed as:

$$\begin{aligned}\mathbb{P}(z_{i\to j} \mid \mathbf{Z}_{-(i\to j)}, \mathbf{G}, \mathbf{P}, \gamma, \mu, \sigma, \lambda, \eta) &= \frac{\mathbb{P}(\mathbf{g}_{ij*} \mid \mathbf{G}_{-(ij*)}, \mathbf{P}, \mathbf{Z}, \lambda, \eta)\mathbb{P}(z_{i\to j} \mid \mathbf{Z}_{-(i\to j)}, \mu, \sigma)}{\int_{z_{i\to j}} \mathbb{P}(\mathbf{g}_{ij*} \mid \mathbf{G}_{-(ij*)}, \mathbf{P}, \mathbf{Z}, \lambda, \eta)\mathbb{P}(z_{i\to j} \mid \mathbf{Z}_{-(i\to j)}, \mu, \sigma) \ \mathrm{d}z_{i\to j}} \\ &\propto \mathbb{P}(\mathbf{g}_{ij*} \mid \mathbf{G}_{-(ij*)}, \mathbf{P}, \mathbf{Z}, \lambda, \eta)\mathbb{P}(z_{i\to j} \mid \mathbf{Z}_{-(i\to j)}, \mu, \sigma) \end{aligned} \tag{22}$$

Where $\mathbf{g}_{ij*} = \{g_{xyz} \in \mathbf{G} : i = x \wedge j = y\}$ denotes the vector of relations in $\mathbf{G}$ from entity e_i to e_j across all predicates, $\mathbf{G}_{-(ij*)} = \mathbf{G} \setminus \mathbf{g}_{ij*}$ is its complement, and $\mathbf{Z}_{-(i\to j)}$ is all the level indicators excluding $z_{i\to j}$. The prior probability for sampling levels, $\mathbb{P}(z_{i\to j} \mid \mathbf{Z}_{-(i\to j)}, \mu, \sigma)$, is drawn from the derived posterior of level memberships in Equation 16 . We follow Blei et al. [61] and use the law of total expectations to obtain the probability of $z_{i\to j}$ realizing level l as the expectation of the size of the stick broken off at the l^{th} break. To do this we define two variables which may be seen as the directed extensions of those introduced in Equation 15:

$$\begin{aligned} \#^{\mathbf{Z}_{-(i\to j)}=l} &= \left|\left\{z_{i\to j} \in \mathbf{Z}_{-(i\to j)} : z_{i\to j} = l\right\}\right| \\ \#^{\mathbf{Z}_{-(i\to j)}>l} &= \left|\left\{z_{i\to j} \in \mathbf{Z}_{-(i\to j)} : z_{i\to j} > l\right\}\right| \end{aligned} \tag{23}$$

With these variables in place, we obtain the prior distribution as follows:

$$\begin{aligned} \mathbb{P}(z_{i\to j} = l \mid \mathbf{Z}_{-(i\to j)}, \mathbf{G}_{-(ij*)}, \mathbf{P}, \mu, \sigma) &= \mathbb{E}\Big[\mathbb{I}(z_{i\to j} = l) \;\Big|\; \mathbf{Z}_{-(i\to j)}, \mu, \sigma\Big] \\ &\overset{(1)}{=} \mathbb{E}\left[\mathbb{E}\left[\mathbb{I}(z_{i\to j} = l) \;\Big|\; v_1, v_2, ..., v_l, \mathbf{Z}_{-(i\to j)}, \mu, \sigma\right]\right] \\ &\overset{(2)}{=} \mathbb{E}\left[\sum_{m=1}^{\infty} v_l \prod_{k=1}^{l-1}(1 - v_k)\mathbb{I}(m = l) \;\Big|\; \mathbf{Z}_{-(i\to j)}, \mu, \sigma\right] \\ &= \mathbb{E}\Big[v^l \mid \mathbf{Z}_{-(i\to j)}, \mu, \sigma\Big] \prod_{k=1}^{l-1} \mathbb{E}\Big[1 - v^k \;\Big|\; \mathbf{Z}_{-(i\to j)}, \mu, \sigma\Big] \\ &\overset{(3)}{=} \frac{\mu\sigma + \#^{\mathbf{Z}_{-(i\to j)}=l}}{\sigma + \#^{\mathbf{Z}_{-(i\to j)}=l} + \#^{\mathbf{Z}_{-(i\to j)}>l}} \prod_{k=1}^{l-1} \frac{(1-\mu)\sigma + \#^{\mathbf{Z}_{-(i\to j)}>k}}{\sigma + \#^{\mathbf{Z}_{-(i\to j)}=k} + \#^{\mathbf{Z}_{-(i\to j)}>k}} \end{aligned} \tag{24}$$

Where (1) is derived by the application of the law of total expectation; (2) is obtained from the probability of drawing level l from the stick breaking process conditioned on the successive draws from the Beta distribution, denoted $v_1, v_2, ..., v_l$, as per Equation 8; and (3) is the expected value of drawing from the Beta distribution conditioned on $\mathbf{Z}_{-(i\to j)}$ as per the level membership posterior obtained in Equation 16. The likelihood, $\mathbb{P}(\mathbf{g}_{ij*} \mid \mathbf{G}_{-(ij*)}, \mathbf{P}, \mathbf{Z}, \lambda, \eta)$, is obtained analogously to entity paths. To aid in this derivation, we define two constants as follows:

$$\begin{aligned}
\#^{c_{pqr}=1}_{-(ijr)} &= \left|\left\{ g_{xyz} \in \mathbf{G}_{-(ijr)} \ : \ \Psi(x,y,z) = c_{pqr} \wedge g_{xyz} = 1 \right\}\right| \\
\#^{c_{pqr}=0}_{-(ijr)} &= \left|\left\{ g_{xyz} \in \mathbf{G}_{-(ijr)} \ : \ \Psi(x,y,z) = c_{pqr} \wedge g_{xyz} = 0 \right\}\right|
\end{aligned} \tag{25}$$

Such that $\#^{c_{pqr}=1}_{-(ijr)}$ and $\#^{c_{pqr}=0}_{-(ijr)}$ capture the number of existing and non-existing relations from communities t_p to t_q not including g_{ijr}. With these in place, we can derive the level indicator likelihood. This process is analogous to the one for entity paths in that we use the Bernoulli distribution for model output and integrate over community relations:

$$\begin{aligned}
&\mathbb{P}(\mathbf{g}_{ij*} \mid \mathbf{G}_{-(ij*)}, \mathbf{P}, \mathbf{Z}, \lambda, \eta) \\
&\quad = \int_{c_{pqr}} \mathbb{P}(\mathbf{g}_{ij*} \mid \mathbf{G}_{-(ij*)}, \mathbf{C}, \mathbf{P}, \mathbf{Z}, \gamma, \mu, \sigma, \lambda, \eta) \mathbb{P}(c_{pqr} \mid \mathbf{C}_{-(pqr)}, \mathbf{G}_{-(ij*)}, \mathbf{P}, \mathbf{Z}, \lambda, \eta) \, \mathrm{d}c_{pqr} \\
&\quad \overset{(1)}{=} \int_{c_{pqr}} \prod_{g_{ijr} \in \mathbf{g}_{ij*}} \mathbb{P}(g_{ijr} \mid \mathbf{G}_{-(ijr)}, \mathbf{C}, \mathbf{P}, \mathbf{Z}, \gamma, \mu, \sigma, \lambda, \eta) \mathbb{P}(c_{pqr} \mid \mathbf{C}_{-(pqr)}, \mathbf{G}_{-(ijr)}, \mathbf{P}, \mathbf{Z}, \lambda, \eta) \, \mathrm{d}c_{pqr} \\
&\quad \overset{(2)}{=} \int_{c_{pqr}} \prod_{g_{ijr} \in \mathbf{g}_{ij*}} \Bigg(\mathrm{Bernoulli}(c_{pqr}, 1 - c_{pqr}) \Bigg) \Bigg(\mathrm{Beta}(\#^{c_{pqr}=1}_{-(ijr)} + \lambda, \#^{c_{pqr}=0}_{-(ijr)} + \eta) \Bigg) \, \mathrm{d}c_{pqr} \\
&\quad = \int_{c_{pqr}} \prod_{g_{ijr} \in \mathbf{g}_{ij*}} \Bigg(c_{pqr}^{g_{ijr}} (1 - c_{pqr})^{1 - g_{ijr}} \Bigg) \Bigg(\frac{c_{pqr}^{\#^{c_{pqr}=1}_{-(ijr)} + \lambda - 1} (1 - c_{pqr})^{\#^{c_{pqr}=0}_{-(ijr)} + \eta - 1}}{\mathrm{B}(\#^{c_{pqr}=1}_{-(ijr)} + \lambda, \#^{c_{pqr}=0}_{-(ijr)} + \eta)} \Bigg) \, \mathrm{d}c_{pqr} \\
&\quad = \prod_{g_{ijr} \in \mathbf{g}_{ij*}} \frac{1}{\mathrm{B}(\#^{c_{pqr}=1}_{-(ijr)} + \lambda, \#^{c_{pqr}=0}_{-(ijr)} + \eta)} \int_{c_{pqr}} c_{pqr}^{g_{ijr} + \#^{c_{pqr}=1}_{-(ijr)} + \lambda - 1} (1 - c_{pqr})^{(1 - g_{ijr}) + \#^{c_{pqr}=0}_{-(ijr)} + \eta - 1} \, \mathrm{d}c_{pqr} \\
&\quad \overset{(3)}{=} \prod_{g_{ijr} \in \mathbf{g}_{ij*}} \frac{\mathrm{B}(\#^{c_{pqr}=1}_{-(ijr)} + g_{ijr} + \lambda, \#^{c_{pqr}=0}_{-(ijr)} + (1 - g_{ijr}) + \eta)}{\mathrm{B}(\#^{c_{pqr}=1}_{-(ijr)} + \lambda, \#^{c_{pqr}=0}_{-(ijr)} + \eta)} \\
&\quad \overset{(4)}{=} \prod_{g_{ijr} \in \mathbf{g}_{ij*}} \frac{\Gamma(\#^{c_{pqr}=1}_{-(ijr)} + g_{ijr} + \lambda) \Gamma(\#^{c_{pqr}=0}_{-(ijr)} + (1 - g_{ijr}) + \eta) \Gamma(\#^{c_{pqr}=1}_{-(ijr)} + \#^{c_{pqr}=0}_{-(ijr)} + \lambda + \eta)}{\Gamma(\#^{c_{pqr}=1}_{-(ijr)} + \#^{c_{pqr}=0}_{-(ijr)} + 1 + \lambda + \eta) \Gamma(\#^{c_{pqr}=1}_{-(ijr)} + \lambda) \Gamma(\#^{c_{pqr}=0}_{-(ijr)} + \eta)} \\
&\quad \overset{(5)}{=} \prod_{g_{ijr} \in \mathbf{g}_{ij*}} \frac{g_{ijr}(\#^{c_{pqr}=1}_{-(ijr)} + \lambda) + (1 - g_{ijr})(\#^{c_{pqr}=0}_{-(ijr)} + \eta)}{\#^{c_{pqr}=1}_{-(ijr)} + \#^{c_{pqr}=0}_{-(ijr)} + \lambda + \eta}
\end{aligned} \tag{26}$$

Where (1) applies the chain rule of probability; (2) utilizes the prior for c_{pqr} obtained in Equation 13; (3) and (4) leverage the integral and Gamma forms of the Beta function as per Equations B.1 and A.3, respectively; and (5) simplifies the preceding equation for computational reason by eliminating the Gamma function as shown in Appendix C. With the prior and likelihood derived in closed form as per Equations 24 and 26, respectively, it is possible to sample level indicators via Equation 22. The time complexity of sampling a level indicator is $\mathcal{O}(|\mathcal{R}|L)$ due to the $|\mathcal{R}|$ calculations that need to be performed at each of the L levels in the hierarchy.

Algorithm 1 Collapsed Gibbs Sampling Procedure for Model Inference

```
Input: Knowledge graph adjacency tensor, $\mathbf{G}$; model hyperparameters, $\gamma$, $\mu$, $\sigma$, $\lambda$, and $\eta$; number of iterations, $iters$
Output: Paths $\mathbf{P}$; level indicators $\mathbf{Z}$;
community relations $\mathbf{C}$
Initialize level indicators using Equation 24
Initialize paths using Equation 18
for $iter = 1, 2, ..., iters$ do
    Update level indicators using Equation 22 if Bernoulli($s_i$)
    Update paths using Equation 17 if Bernoulli($s_i$)
end for
Obtain final level indicators using Equation 22
Obtain final paths using Equation 17
```

4.2.5 Sampling Procedure

Having marginalized out community relations and level memberships as well as derived the sampling equations for entity paths and level indicators, it is possible to perform collapsed Gibbs sampling by iteratively sampling from the remaining variables' full conditional distributions. This process has a time complexity of $\mathcal{O}(|\mathcal{E}|^2|\mathcal{R}|L + |\mathcal{E}|^3|\mathcal{R}|L)$ for each iteration of the sampler where the former and latter terms are derived from the sampling complexities of the level indicators and entity paths, respectively. This makes the inference scheme infeasible for large-scale datasets. We respond to this issue by modifying one of the characteristics of collapsed Gibbs sampling, namely that samples are obtained in equal proportions. In its original formulation, one iteration of the sampler samples $|\mathcal{E}|^2$ level indicators and $|\mathcal{E}|$ entity paths. One of the assumptions underlying this process is that the relative importance of all samples is the same. Such an assumption may be ill-adapted for knowledge graphs which are oftentimes sparse in their adjacency tensors and whose entities exhibit highly imbalanced relation densities. In this regard, the placement of highly connected entities will have a disproportionate effect on the model likelihood and therefore the induced hierarchy as well. Preferentially sampling these entities may result in faster arrival at a distribution from which we can obtain output samples. Consider, for instance, a knowledge graph with the entities `Thing` and `Henry Ford`. Assuming that `Thing` has a higher relation density than `Henry Ford`, its proper placement in the hierarchy may be more critical for model output than `Henry Ford`. With this in mind, we propose a stochastic sampling scheme in which samples are drawn for an entity in proportion to their probability of interacting with other entities. Specifically, we introduce a sampling probability, denoted s_i for entity e_i, which specifies the chance of sampling a variable for the corresponding entity in an iteration of the collapsed Gibbs sampler. This probability is calculated for each entity as the fraction of entities in the knowledge graph which have fewer relations than itself. Such as formulation ensures that $1 \geqslant s_i > 0$ which allows s_i to serve as the parameter of a Bernoulli distribution to indicate whether a variable will get sampled in the current iteration of the Gibbs sampler.

After the Gibbs sampler has been burned in, it is necessary to obtain final samples to induce a hierarchical clustering. We take multiple samples to account for the spread in the posterior distribution. A consequence of this is that samples may differ and need to be aggregated to produce a final result. In this regard, we take the mode over the final samples to arrive at a final hierarchy. The Gibbs sampling procedure is summarized in Algorithm 1.

5 Evaluation

The evaluation of our model is split into two parts: quantitative and qualitative. The quantitative evaluation provides objective measures of model performance whereas the qualitative evaluation assesses our model through illustrations and subjective analysis of the results. For both types of evaluations, our model first had to be inferred before final samples could be drawn. In this regard, we trained our model on three datasets using 200 burn-in samples using hyperparameters chosen by assessing the model's log likelihood. After burn-in, ten final samples were obtained by discarding all but the third of successive samples to account for autocorrelation between samples. All models we trained to a depth of $L = 4$. Furthermore, the model was trained five times for each dataset to account for stochasticity in the inference process. The implementation of our method as well as the datasets necessary to recreate our evaluation has been made publicly available on GitHub [2].

5.1 Datasets

Our model was evaluated on three datasets: Synthetic Binary Tree, FB15k-237, and Wikidata. What follows is a brief description of each dataset as well as how it was generated.

5.1.1 Synthetic Binary Tree

The Synthetic Binary Tree (SBT) dataset was synthetically generated to capture our model's ability to separate communities at the lowest level in the hierarchy. The generative process first constructed a binary tree with a depth of four, assigned entities to communities, and sampled relations for each entity pair. All entities were assigned uniformly to communities on the lowest level of the hierarchy, resulting in 25 entities per leaf community. The sampling probability for each entity pair was determined by the level of their lowest common ancestor. Specifically, sampling probabilities of 0, 0.1, 0.4, and 0.6 were used for levels 0, 1, 2, and 3, respectively. Two entities belonging to the same community have a sampling probability of 1 and are thus always related. The dataset was generated for two predicates which shared the aforementioned sampling probabilities. We note that even though these probabilities are identical, they do not result in a dataset in which entity relations are identical across predicates. The generative process yielded a dataset of 55880 triples, 400 entities, and 2 predicates.

5.1.2 FB15k-237

The FB15k-237 dataset [64] is a subset of the FB15k dataset [34], created by removing redundant and inverse triples. The original FB15k dataset is in turn a subset of a 2013 version of Freebase, from which triples were queried. The FB15k-237 dataset is comprised of 272115 triples, 14541 entities, and 237 predicates thus presenting a computation challenge to our model if modelled in whole. To address this issue, we generated a subset of the data and derived ground truth community labels in an approach insipred by Jain et al. [65]. Specifically, entities were mapped to the WordNet taxonomy [66] through the *sameAs* predicate, which relates entities from Freebase and YAGO. Triples were then extracted to contain subjects from the sets provided in Zhang et al. [13]. This process yielded a subset of the data containing 103550 triples, 10018 entities, and 190 predicates. Finally, the subset was reduced even further by extracting only the triples relating to footballers, pianists, journalists, politicians, and scientists as per the identifiers `/m/05vyk`, `/m/06q2q`, `/m/0gl2ny2`, `/m/0fj9f`, `/m/0d8qb` on the predicate `/people/person/profession`. This final step resulted in a dataset with 2499 triples, 1142 entities, and 79 predicates.

5.1.3 Wikidata

The Wikidata dataset was generated by querying Wikidata for triples relating to people and locations. Specifically, artists and footballers corresponding to Wikidata identifiers `wd:Q1028181` and `wd:Q937857` respectively were extracted. These entities were then filtered to having been born in cities in four countries: Germany, the United Kingdom, Canada, and the United States of America. Furthermore, the knowledge graph was reduced to the following predicates: instance of, place of birth, citizen of, occupation, country, and located in which are represented by the identifiers `wdt:P31`, `wdt:P19`, `wdt:P27`, `wdt:P106`, `wdt:P17`, and `wdt:P131`, respectively. Finally, the tripleset was further reduced to yield 2525 triples, 716 entities, and 6 predicates.

5.2 Quantitative Evaluation

In our quantitative evaluation, we first analyzed the quality of our learned hierarchical clustering by calculating two clustering quality metrics at each level of the hierarchy: the Adjusted Rand Index (ARI) [67] and Normalized Mutual Information (NMI) [68]. This type of evaluation jointly assesses the quality of the learned community hierarchy as well as the membership of entities to communities. The ARI is an adjustment to the commonly used Rand Index (RI) [69], corrected to account for chance. Specifically, chance is factored in by calculating the expected

[2] https://www.github.com/mpietrasik/hb

	SBT		FB15k-237		Wikidata	
Method	ARI	NMI	ARI	NMI	ARI	NMI
Level 1	0.3055	0.4855	0.5326	0.6646	0.8411	0.7991
	±0.0685	±0.1013	±0.1308	±0.0702	±0.2980	±0.1581
Level 2	0.5895	0.7826	0.3492	0.5083	0.8057	0.7232
	±0.2826	±0.1434	±0.2044	±0.1175	±0.2839	±0.1410
Level 3	0.7279	0.8882	0.2851	0.4329	0.4255	0.5880
	±0.1656	±0.0621	±0.1993	±0.1030	±0.2749	±0.1367
Level 4	0.8337	0.9319	0.1964	0.5334	0.3812	0.4980
	±0.1032	±0.0357	±0.0438	±0.0288	±0.2500	±0.1309
Overall	0.6141	0.7721	0.3408	0.5348	0.6134	0.6521
	±0.2577	±0.1988	±0.1867	±0.1145	±0.3341	±0.1770

Table 1

ARI and MNI scores (mean ± standard deviation) of our model on the SBT, FB15k-237 and Wikidata datasets.

	SBT		FB15k-237		Wikidata	
Method	ARI	NMI	ARI	NMI	ARI	NMI
RDF2VEC						
k-means	0.8060	0.8928	0.0109	0.1402	0.2672	0.2918
	±0.1845	±0.0707	±0.0929	±0.1052	±0.1582	±0.1040
Agglomerative	0.8750	0.9317	0.0461	0.1532	0.4674	0.5287
	±0.1254	±0.0575	±0.0860	±0.1435	±0.3281	±0.2052
DBSCAN	0.5549	0.6904	0.1468	0.2293	0.3831	0.3698
	±0.4576	±0.3032	±0.1291	±0.0561	±0.2343	±0.0935
Spectral	0.6175	0.7590	-0.0014	0.0347	0.0918	0.1021
	±0.3540	±0.2924	±0.0082	±0.03129	±0.0636	0.0297
TransE						
k-means	0.9851	0.9958	**0.3559**	0.4334	**0.7427**	0.6504
	±0.0334	±0.0066	±0.0776	±0.1096	±0.1953	±0.2468
Agglomerative	**1.0000**	**1.0000**	0.1362	0.3107	0.3799	0.3650
	±0.0000	±0.0000	±0.1379	±0.1104	±0.4037	±0.3780
DBSCAN	0.8899	0.9665	0.2768	0.2582	0.2418	0.3128
	±0.1213	±0.03829	±0.1616	±0.0728	±0.1355	±0.1143
Spectral	**1.0000**	**1.0000**	0.1400	0.2509	0.2778	0.3296
	±0.0000	±0.0000	±0.1920	±0.2418	±0.1541	±0.0153
Our method	0.6141	0.7721	0.3408	**0.5348**	0.6134	**0.6521**
	±0.2577	±0.1988	±0.1867	±0.1145	±0.3341	±0.1770

Table 2

ARI and MNI scores (mean ± standard deviation) of our model on the SBT, FB15k-237 and Wikidata datasets as compared with baseline approaches.

RI given a random clustering and measuring the obtained clustering's deviation. Specifically, given an obtained entity clustering $\mathcal{C} = \{\mathcal{C}_1, \mathcal{C}_2, \dots, \mathcal{C}_o\}$ and the ground truth clustering $\mathcal{C}^* = \{\mathcal{C}^*_1, \mathcal{C}^*_2, \dots, \mathcal{C}^*_t\}$, the ARI is calculated as follows:

$$ARI = \frac{\sum_{\mathcal{C}_i \in \mathcal{C}} \sum_{\mathcal{C}^*_j \in \mathcal{C}^*} \binom{\#_{ij}}{2} - \binom{|\mathcal{E}|}{2}^{-1} \Big(\sum_{\mathcal{C}_i \in \mathcal{C}} \binom{\#_i}{2} \sum_{\mathcal{C}^*_j \in \mathcal{C}^*} \binom{\#^*_j}{2} \Big)}{2^{-1} \Big(\sum_{\mathcal{C}_i \in \mathcal{C}} \binom{\#_i}{2} + \sum_{\mathcal{C}^*_j \in \mathcal{C}^*} \binom{\#^*_j}{2} \Big) - \binom{|\mathcal{E}|}{2}^{-1} \Big(\sum_{\mathcal{C}_i \in \mathcal{C}} \binom{\#_i}{2} \sum_{\mathcal{C}^*_j \in \mathcal{C}^*} \binom{\#^*_j}{2} \Big)} \qquad (27)$$

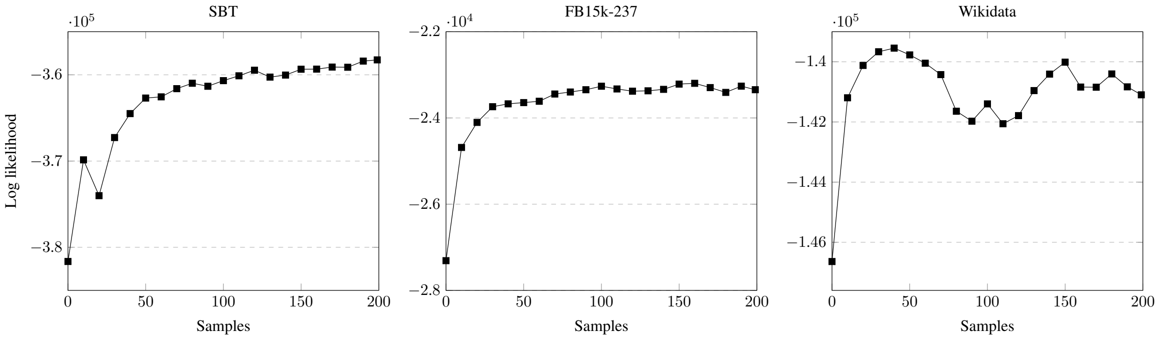


Fig. 8. Plots of average log likelihood of our model across burn in samples on three datasets.

where $\#_{ij} = |\mathcal{C}_i \cap \mathcal{C}_j^*|$ is the number of entities in common between a ground truth and obtained cluster pair; $\#_i = \sum_{\mathcal{C}_j^* \in \mathcal{C}^*} |\mathcal{C}_i \cap \mathcal{C}_j^*|$ is the total number of entities in obtained cluster $\mathcal{C}_i$; and $\#_j^* = \sum_{\mathcal{C}_i \in \mathcal{C}} |\mathcal{C}_i \cap \mathcal{C}_j^*|$ is the total number of entities in ground truth cluster $\mathcal{C}_j^*$. The NMI is a normalized extension of the Mutual Information (MI) score which quantifies the information gained about the obtained clustering given the ground truth clusters. The normalization of the MI score ensures the result is in the range $[0, 1]$ thereby allowing for its comparison against clusterings of different sizes. Utilizing the notation defined earlier, we define MI and NMI as follows:

$$NMI = \frac{\sum_{\mathcal{C}_i \in \mathcal{C}} \sum_{\mathcal{C}_j^* \in \mathcal{C}^*} \frac{|\mathcal{C}_i \cap \mathcal{C}_j^*|}{|\mathcal{E}|} \log\left(\frac{|\mathcal{E}||\mathcal{C}_i \cap \mathcal{C}_j^*|}{|\mathcal{C}_i||\mathcal{C}_j^*|}\right)}{\text{mean}\left(-\sum_{\mathcal{C}_i \in \mathcal{C}} \frac{|\mathcal{C}_i|}{|\mathcal{E}|} \log\left(\frac{|\mathcal{C}_i|}{|\mathcal{E}|}\right), -\sum_{\mathcal{C}_j^* \in \mathcal{C}^*} \frac{|\mathcal{C}_j^*|}{|\mathcal{E}|} \log\left(\frac{|\mathcal{C}_j^*|}{|\mathcal{E}|}\right)\right)} \tag{28}$$

For both the ARI and NMI, higher scores indicate a clustering of higher quality. We summarize the results of our clustering as per these two metrics in Table 1.

In general, the results indicate that our model is capable of learning a coherent community hierarchy on each of the three datasets tested. Perhaps unsurprisingly, communities at higher levels in the hierarchy are judged as of higher quality as per the two evaluation metrics. This is because the task of clustering entities at higher levels is simpler as the communities are less fine-grained. For instance, on the FB15k-237 dataset, clustering at level 1 requires the distinction between `Place` and `Person` whereas level 4 requires the distinction between `AmericanFootballPlayer` and `IceHockeyPlayer`. We note that the SBT dataset is an exception to this. This is likely due to the nature of the dataset wherein entity relations are drawn at higher proportions between neighbouring communities at lower levels of the hierarchy. In this sense, the claim made before gets inverted and it is easier to assign communities at lower levels in the hierarchy. We also compared our model against embedding and clustering methods used in conjunction. Specifically, we first embedded each of the knowledge graphs using the RDF2VEC and TransE embedding methods. Afterwards, we applied four clustering methods: *k*-means, Agglomerative, DBSCAN, and Spectral. These results are summarized in Table 2 and indicate comparable or superior performance to baselines.

We can also analyze the results of the complete log likelihood as a function of the number of Gibbs samples taken in the inference process. Indeed, while this does not provide us with information about the quality of the obtained results, it does verify the inference process itself. Specifically, we expect to see the log likelihood of our model to rise given more burn-in samples of the Gibbs sampler. This suggests that the likelihood of generating the knowledge graph given the current state of the sampler is increasing and learning is taking place. We can see this rise in Figure 8 which plots the complete log likelihoods of our model across Gibbs samples for the three datasets. We note a dip in log likelihoods on the SBT and Wikidata datasets. This is likely due to the sampler being temporarily stuck in a local minimum before leaving that area in the sample space. The underperformance on the SBT dataset compared to the baseline approaches is due to our method's underperformance on simple datasets, largely attributed to its stochastic

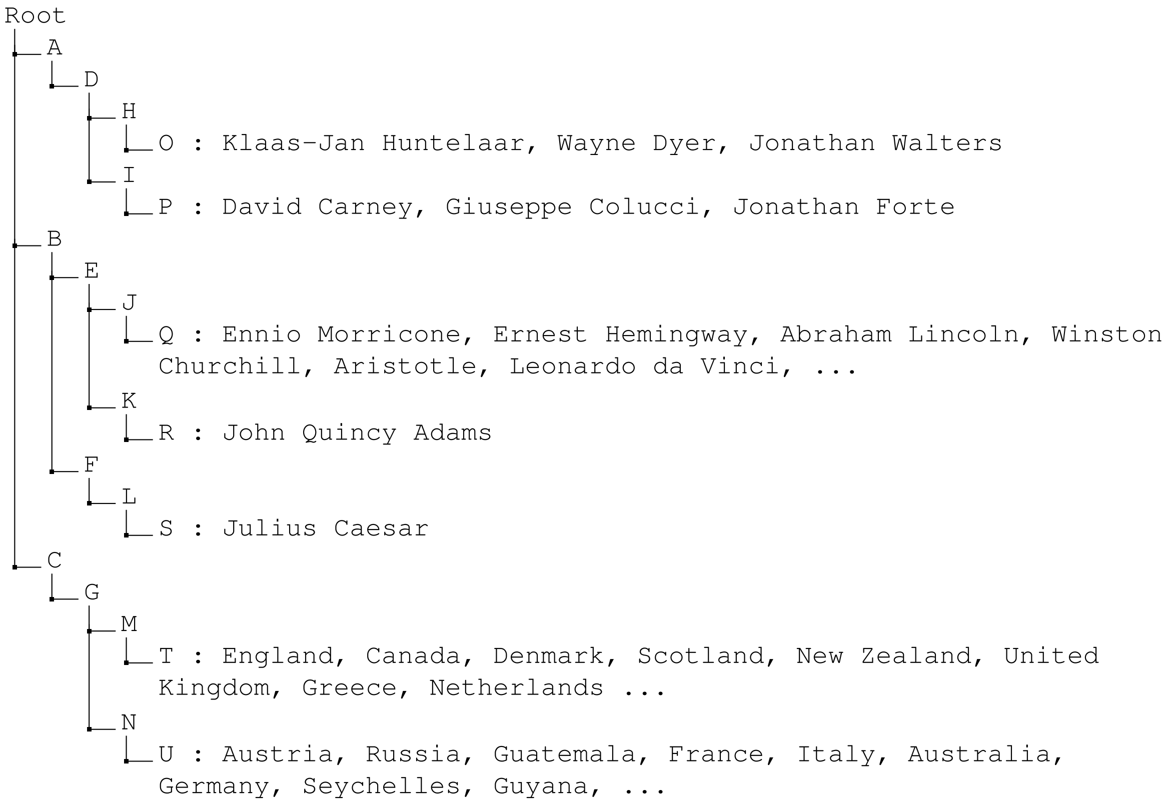

Root
A
D
H
O : Klaas-Jan Huntelaar, Wayne Dyer, Jonathan Walters
I
P : David Carney, Giuseppe Colucci, Jonathan Forte
B
E
J
Q : Ennio Morricone, Ernest Hemingway, Abraham Lincoln, Winston Churchill, Aristotle, Leonardo da Vinci, ...
K
R : John Quincy Adams
F
L
S : Julius Caesar
C
G
M
T : England, Canada, Denmark, Scotland, New Zealand, United Kingdom, Greece, Netherlands ...
N
U : Austria, Russia, Guatemala, France, Italy, Australia, Germany, Seychelles, Guyana, ...

Fig. 9. Excerpt of our induced hierarchy on the FB15k-237 dataset. Entities in communities `O` and `P` are footballers. `Ennio Morricone` is a pianist. `Ernest Hemmingway` is a journalist. `Abraham Lincoln`, `Winston Churchill`, `John Quincy Adams`, and `Julius Caesar` are politicians. `Aristotle` and `Leonardo da Vinci` are scientists. The entities in communities `T` and `U` are countries.

nature. Recall that after burn-in, final samples are drawn and unless the log-likelihood is sufficiently high, these samples will contain suboptimal allocations. These samples are necessary for Gibbs sampling but result in poorer quantitative performance. We hypothesize that drawing more burn-in samples would stabilize the final samples and produce better results. This is supported by the log-likelihoods in Figure 8 but, to establish consistency between the three datasets, was not explored further.

5.3 Qualitative Evaluation

In our qualitative evaluation, we leverage the qualitative attributes possessed by a useful taxonomy as identified by Nickerson et al. [70]. Although these attributes were proposed in the context of taxonomy development, we make use of them in our work as the task of hierarchical clustering shares many of the underlying evaluation principles. Indeed, a taxonomy is implicitly induced using our method, although never explicitly labelled. The proposed taxonomy attributes are as follows: concise, robust, comprehensive, extendable, and explanatory. For each of these attributes, we provide a brief description extended to hierarchical clustering and use it to evaluate our model.

- A concise clustering is limited in both the number of obtained clusters and the semantic diversity of the entities that constitute each cluster. In our method, this is largely regulated by the hyperparameters γ, allowing for control over the number of clusters obtained, and the interplay between λ and σ, regulating the intracluster density. In practice, however, the hyperparameter tuning required is fickle. Figure 9 demonstrates this through an excerpt of the results obtained on the FB15k-237 dataset. Singleton communities `R` and `S` are both composed of politician entities and, in keeping with the conciseness attribute, warrant a merger. Community `Q` on the other hand is composed of a heterogeneous set of entities and requires splitting. It is concise, but, as we shall

```
Root
|__A
|  |__E
|     |__J
|        |__Q : Adrian Kleinbergen, Isaac Rosenberg, Lynd Ward, John William
|        |   Inchbold, Walt Disney, Helen Frankenthaler, ...
|        |__R : Charles Krafft, Kenneth Armitage, Beth Hart, ...
|__B
|  |__F
|     |__K
|        |__S : Owen Hargreaves, Thomas Hitzlsperger, Franz Beckenbauer, Andy
|        |   Welsh, Paul Breitner, ...
|        |__T : Adam Smith
|__C
|  |__H
|     |__M
|     |  |__V : Dortmund, Toronto, Ottawa, Munich, ...
|     |__N
|        |__W : Edmonton, Manchester
|__D
   |__I
      |__O
      |  |__X : England, Sweden, Netherlands, Uruguay, Tunisia, ...
      |  |__Y : Spain, Malta, Japan, Norway, Great Britain, Finland,
      |      Paraguay, Kingdom of Saxony
      |__P
         |__Z : Poland, France
```

Fig. 10. Excerpt of our induced hierarchy on the Wikidata dataset. Entities in communities Q and R are painters, in communities S and T are footballers, in communities V and W are cities, and in communities X, Y, and Z are countries as defined in Wikidata.

elaborate upon later, it is not robust. Thus the direction in which to adjust the hyperparameters is not clear. The induced hierarchy also suffers in terms of conciseness at higher levels in the hierarchy. Communities A and B are both comprised of people and require a merger at their level. This issue can largely be explained by the data itself. Footballers in our dataset have structural properties which differ them from the persons clustered in the community B subtree. In addition to sharing the same profession, they belong to football teams, have football specific triples such as the position they play, and are more likely than non-footballer persons in the dataset to have information relating to physical characteristics such as height and weight. In contrast, non-footballer persons have less structural similarities, even within members of their own professions. For instance, even though all scientists have their scientific contributions, this is not reflected in the data as uniformly. The Wikidata dataset – an excerpt of which is displayed in Figure 10 – is simpler than FB15k-237 due to there being only two professions with largely disjoint neighbourhoods. The induced hierarchy is not concise, however, in the splitting of persons and places. Specifically, this happens at the first level where persons are split into communities A and B and places into communities C and D. We note, however, the high ARI and NMI scores at this level despite this error. A closer inspection reveals that not all runs of our model encounter this issue, thus the scores are higher than they would be had they been measured only on the hierarchy presented in the excerpt.

- Robustness in clustering refers to "maximizing both within-group homogeneity and between-group heterogeneity, [making] groups that are as distinct (nonoverlapping) as possible, with all members within a group being as alike as possible [71]." We see robustness as an issue when examining the non-footballer professions in Figure 9. Pianists, journalists, politicians, and scientists are not sufficiently semantically homogenous to

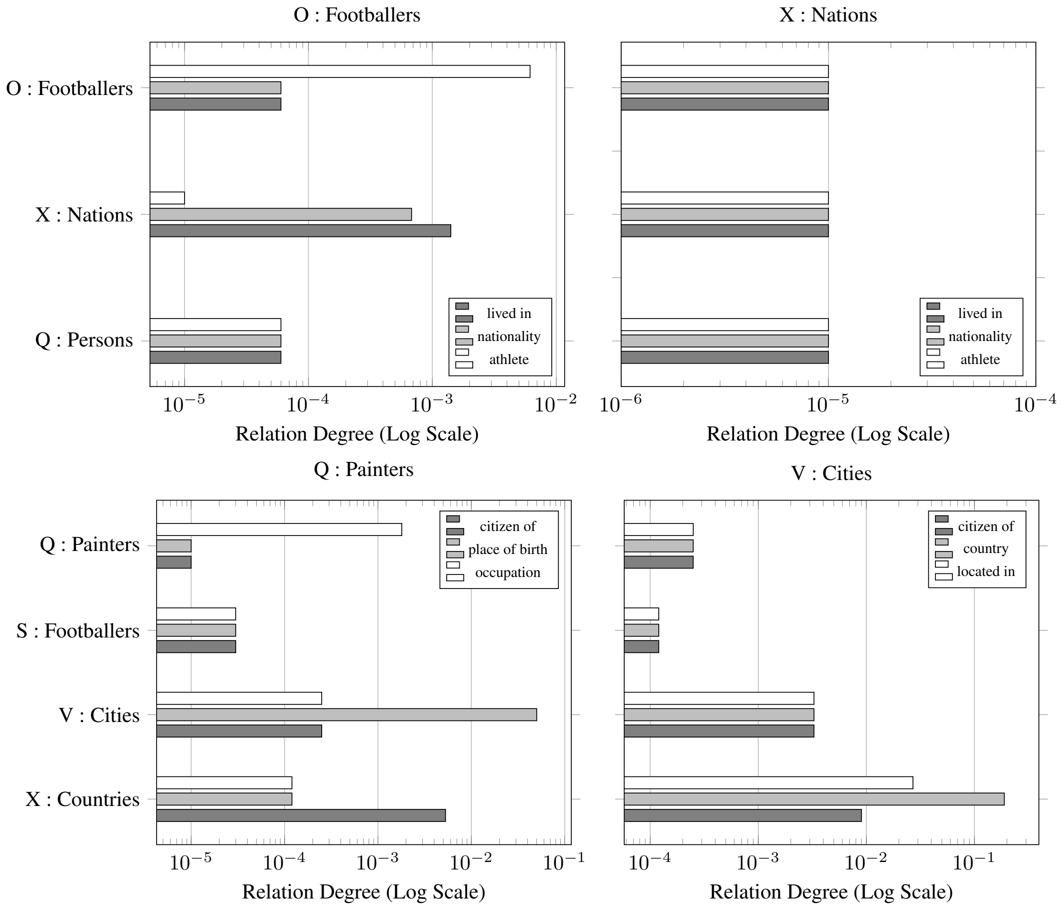


Fig. 11. Plots of learned community relations for selected outgoing predicates for the FB15k-237 and Wikidata datasets. Specifically we showcased community `O` (Footballers) and community `T` (Nations) outgoing relations for the FB15k-237 dataset (top) and community `Q` (Painters) and community `V` (Cities) outgoing relations for the Wikidata dataset.

warrant their inclusion in community `Q`. Another issue is the splitting of nations into communities `O` and `P` as there is no evident geographic, social, political, or economic distinction made in the clustering. A deeper look into the data also reveals no apparent structural differences between communities `M` and `N`. It is likely, therefore, that this issue can be due to model inaccuracy and arrival at a suboptimal state after inference. The Wikidata dataset also suffers from unsubstantiated splits at the lowest levels as seen in communities `R`, `T`, and `Y`. This issue may be the result of the small size of the data and the model's resultant sensitivity to the relational information provided. For instance, close to half of the entities in this dataset have first order neighborhoods of one or two entities, giving our model little to learn from.

- A taxonomy is comprehensive if it can classify all entities in the domain. Clustering, including the hierarchical clustering obtained by our method, are induced empirically from the data and thus necessarily classify all entities into communities. As such, our generative model is comprehensive.
- An extendable clustering is one that allows for the dynamic inclusion of additional information and, in the hierarchical case, the structural change and adaptability to incoming information. In these two respects, our method is highly extendable. Indeed, the Gibbs sampling process itself requires the removal of entities from the hierarchy before they are resampled. Each resampling not only has the potential to change the community assignment of the entity but also to change the structure of the hierarchy itself. Due to the infinite formulation of the nCRP and stick breaking process, there is no prior constraint on the structure.

- The final qualitative attribute identified in a useful taxonomy is that it is explanatory. In this regard, the taxonomy should provide useful explanations about the objects it is organizing. In the context of hierarchical clustering, these explanations could, for instance, take the form of community labels. Although our model does not assign labels to communities, they can be ascertained by examining the type information of their constituent entities. For instance in Figure 9 the FB15k-237 communities `O`, `X`, and `P` correspond to footballers, nations, and persons and the Wikidata communities `Q`, `S`, `V`, and `X` correspond to painters, footballers, cities, and countries. An explanatory advantage of our method is that it infers community relations as part of the generative process. A fragment of these relations is conveyed in Figure 11. The results indicate community relations which are largely expected. For instance, on the FB15k-237 dataset, footballers are much more likely to be related to nations by predicates `nationality` and `lived in` than `athlete`. Furthermore, we see that nations are equally unlikely to be the subjects with the other communities or predicates such as lived in, nationality, and athlete. On the Wikidata dataset, we likewise see explainable results. Painters, for instance, are likely to be related to cities by place of birth and countries by nationality. They are unlikely to relate to other painters and footballers on these predicates.

6 Conclusion

In this paper, we demonstrated the use of stochastic blockmodels for learning hierarchies from knowledge graphs in an approach that is, to the best of our knowledge, the first to marry these two fields in an academic setting. To this end, we proposed a model which leverages the nCRP and stick breaking process to generate a community hierarchy composed of a knowledge graph's constituent entities. The model is defined non-parametrically and thus makes no assumptions about its structure, allowing it to adapt to the knowledge graph and potentially induce an infinite number of communities on an infinite number of levels. In addition to the model itself, a Markov chain Monte Carlo scheme leveraging collapsed Gibbs sampling was devised for posterior inference of the model's parameters. The model was evaluated on three datasets using quantitative and qualitative analysis to demonstrate its effectiveness in learning coherent community hierarchies on both synthetic and real-world data. The qualitative analysis made use of attributes commonly employed in taxonomy evaluation, presenting a novel and principled approach for qualitative analysis of hierarchical clusterings. Future work will investigate scalability when applying our model – and indeed stochastic blockmodels more generally – to knowledge graphs. As discussed earlier, the time complexities of the inference schemes for these models usually do not allow for scaling to the types of large knowledge bases that are encountered in real-world applications. The inference scheme proposed in this work is one such instance however several methods exist in the literature more scalable than collapsed Gibbs sampling. An example of such a method is variational inference, a scheme that uses the evidence lower bound to guide the inference process and obtain the posterior distribution. This method is generally faster than Gibbs sampling and, although not asymptotically exact, produces similar results [72]. The challenge with this approach is that its optimization equations are more difficult to solve as compared to Markov chain Monte Carlo methods. Despite this, several works have already successfully applied variational inference to probabilistic graphical models. Indeed, the original inference scheme for the Mixed Membership Stochastic Blockmodel leveraged variational inference. Furthermore, Blei and Jordan [73] provided a variational inference scheme for Dirichlet processes and a variational inference scheme for the nCRP was proposed in Wang and Blei [74]. Departing from the variational approach, Chen et al. [75] propose an evolution of the Gibbs sampling algorithm for the nCRP with partially collapsed Gibbs sampling. This approach resulted in a time increase of over two magnitudes over the classic Gibbs sampling approach. Another line of approach to increase the scalability of stochastic blockmodels is to devise a model which does not require sampling all $|\mathcal{E}|^2|\mathcal{R}|$ relations in the knowledge graph directly. To this end, the Bernoulli-Poisson link function has been applied successfully to simple graphs [76–78]. These methods eliminate the need for quadratic time relation sampling and instead rely on density based sampling which is less computationally demanding, especially on sparse networks. Given that most knowledge graphs are highly sparse, applying such an approach appears promising. The work presented in this paper provides evidence for further research in this direction in the Semantic Web community.

References

[1] solidIT, DBMS popularity broken down by database model, 2022. https://db-engines.com/en/ranking_categories.

[2] B. Jacob, You need to be thinking in knowledge graphs, 2021. https://www.forbes.com/sites/forbestechcouncil/2021/09/20/you-need-to-be-thinking-in-knowledge-graphs/.

[3] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P.N. Mendes, S. Hellmann, M. Morsey, P. Van Kleef, S. Auer et al., DBpedia–a large-scale, multilingual knowledge base extracted from Wikipedia, *Semantic Web* **6**(2) (2015), 167–195.

[4] T. Pellissier Tanon, G. Weikum and F. Suchanek, Yago 4: A reason-able knowledge base, in: *European Semantic Web Conference*, Springer, 2020, pp. 583–596.

[5] D. Vrandečić and M. Krötzsch, Wikidata: a free collaborative knowledgebase, *Communications of the ACM* **57**(10) (2014), 78–85.

[6] R. Xie, Z. Liu, M. Sun et al., Representation Learning of Knowledge Graphs with Hierarchical Types., in: *IJCAI*, Vol. 2016, 2016, pp. 2965–2971.

[7] Z. Zhang, J. Cai, Y. Zhang and J. Wang, Learning hierarchy-aware knowledge graph embeddings for link prediction, in: *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34, 2020, pp. 3065–3072.

[8] P.W. Holland, K.B. Laskey and S. Leinhardt, Stochastic blockmodels: First steps, *Social networks* **5**(2) (1983), 109–137.

[9] X. Wang, W. Yang, Y. Yang, Y. He, J. Zhang, L. Wang and L. Hu, PPISB: A Novel Network-Based Algorithm of Predicting Protein-Protein Interactions With Mixed Membership Stochastic Blockmodel, *IEEE/ACM Transactions on Computational Biology and Bioinformatics* (2022).

[10] T.M. Sweet, Modeling social networks as mediators: A mixed membership stochastic blockmodel for mediation, *Journal of Educational and Behavioral Statistics* **44**(2) (2019), 210–240.

[11] E.M. Airoldi, D. Blei, S. Fienberg and E. Xing, Mixed membership stochastic blockmodels, *Advances in neural information processing systems* **21** (2008).

[12] Q. Ho, A. Parikh, L. Song and E. Xing, Multiscale community blockmodel for network exploration, in: *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, JMLR Workshop and Conference Proceedings, 2011, pp. 333–341.

[13] Y. Zhang, M. Pietrasik, W. Xu and M. Reformat, Hierarchical Topic Modelling for Knowledge Graphs, in: *European Semantic Web Conference*, Springer, 2022, pp. 270–286.

[14] K. Nowicki and T.A.B. Snijders, Estimation and prediction for stochastic blockstructures, *Journal of the American statistical association* **96**(455) (2001), 1077–1087.

[15] D.J. Aldous, Exchangeability and related topics, in: *École d'Été de Probabilités de Saint-Flour XIII—1983*, Springer, 1985, pp. 1–198.

[16] C. Kemp, J.B. Tenenbaum, T.L. Griffiths, T. Yamada and N. Ueda, Learning systems of concepts with an infinite relational model, in: *AAAI*, Vol. 3, 2006, p. 5.

[17] F.A. Saad and V.K. Mansinghka, Hierarchical infinite relational model, in: *Uncertainty in Artificial Intelligence*, PMLR, 2021, pp. 1067–1077.

[18] X. Fan, L. Cao and R.Y. Da Xu, Dynamic infinite mixed-membership stochastic blockmodel, *IEEE transactions on neural networks and learning systems* **26**(9) (2014), 2072–2085.

[19] M. Berlingerio, M. Coscia and F. Giannotti, Finding redundant and complementary communities in multidimensional networks, in: *Proceedings of the 20th ACM international conference on Information and knowledge management*, 2011, pp. 2181–2184.

[20] M. Barigozzi, G. Fagiolo and G. Mangioni, Identifying the community structure of the international-trade multi-network, *Physica A: statistical mechanics and its applications* **390**(11) (2011), 2051–2066.

[21] S. Paul and Y. Chen, Consistent community detection in multi-relational data through restricted multi-layer stochastic blockmodel, *Electronic Journal of Statistics* **10**(2) (2016), 3807–3870.

[22] C. De Bacco, E.A. Power, D.B. Larremore and C. Moore, Community detection, link prediction, and layer interdependence in multilayer networks, *Physical Review E* **95**(4) (2017), 042317.

[23] M. Pietrasik and M. Reformat, Neural Blockmodeling for Multilayer Networks, in: *2021 International Joint Conference on Neural Networks (IJCNN)*, IEEE, 2021, pp. 1–8.

[24] C. Lee and D.J. Wilkinson, A review of stochastic block models and extensions for graph clustering, *Applied Network Science* **4**(1) (2019), 1–50.

[25] J. Völker and M. Niepert, Statistical schema induction, in: *Extended Semantic Web Conference*, Springer, 2011, pp. 124–138.

[26] M. Pietrasik and M. Reformat, A simple method for inducing class taxonomies in knowledge graphs, in: *European Semantic Web Conference*, Springer, 2020, pp. 53–68.

[27] M. Pietrasik and M. Reformat, Path Based Hierarchical Clustering on Knowledge Graphs, *arXiv preprint arXiv:2109.13178* (2021).

[28] P. Heymann and H. Garcia-Molina, Collaborative creation of communal hierarchical taxonomies in social tagging systems, Technical Report, Stanford, 2006.

[29] P. Schmitz, Inducing ontology from flickr tags, in: *Collaborative Web Tagging Workshop at WWW2006, Edinburgh, Scotland*, Vol. 50, 2006, p. 39.

[30] S. Wang, T. Wang, X. Mao, G. Yin and Y. Yu, A hybrid approach for tag hierarchy construction, in: *International Conference on Software Reuse*, Springer, 2018, pp. 59–75.

[31] J.X. Chen and M.Z. Reformat, Learning categories from linked open data, in: *International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*, Springer, 2014, pp. 396–405.

[32] S.K. Mohamed, Unsupervised Hierarchical Grouping of Knowledge Graph Entities, *arXiv preprint arXiv:1908.07281* (2019).

[33] A. Bonifati, S. Dumbrava and N. Mir, Hierarchical Clustering for Property Graph Schema Discovery., in: *EDBT*, 2022, pp. 2–449.

[34] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston and O. Yakhnenko, Translating embeddings for modeling multi-relational data, *Advances in neural information processing systems* **26** (2013).

[35] Z. Wang, J. Zhang, J. Feng and Z. Chen, Knowledge graph embedding by translating on hyperplanes, in: *Proceedings of the AAAI conference on artificial intelligence*, Vol. 28, 2014.

[36] Y. Lin, Z. Liu, M. Sun, Y. Liu and X. Zhu, Learning entity and relation embeddings for knowledge graph completion, in: *Twenty-ninth AAAI conference on artificial intelligence*, 2015.

[37] G. Ji, S. He, L. Xu, K. Liu and J. Zhao, Knowledge graph embedding via dynamic mapping matrix, in: *Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (volume 1: Long papers)*, 2015, pp. 687–696.

[38] M. Nickel, V. Tresp and H.-P. Kriegel, A three-way model for collective learning on multi-relational data, in: *ICML*, 2011.

[39] B. Yang, W.-t. Yih, X. He, J. Gao and L. Deng, Embedding entities and relations for learning and inference in knowledge bases, *arXiv preprint arXiv:1412.6575* (2014).

[40] S.M. Kazemi and D. Poole, Simple embedding for link prediction in knowledge graphs, *Advances in neural information processing systems* **31** (2018).

[41] I. Balažević, C. Allen and T.M. Hospedales, Tucker: Tensor factorization for knowledge graph completion, *arXiv preprint arXiv:1901.09590* (2019).

[42] M. Schlichtkrull, T.N. Kipf, P. Bloem, R.v.d. Berg, I. Titov and M. Welling, Modeling relational data with graph convolutional networks, in: *European semantic web conference*, Springer, 2018, pp. 593–607.

[43] T. Dettmers, P. Minervini, P. Stenetorp and S. Riedel, Convolutional 2d knowledge graph embeddings, in: *Proceedings of the AAAI conference on artificial intelligence*, Vol. 32, 2018.

[44] D.Q. Nguyen, T.D. Nguyen, D.Q. Nguyen and D. Phung, A novel embedding model for knowledge base completion based on convolutional neural network, *arXiv preprint arXiv:1712.02121* (2017).

[45] T. Vu, T.D. Nguyen, D.Q. Nguyen, D. Phung et al., A capsule network-based embedding model for knowledge graph completion and search personalization, in: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019, pp. 2180–2189.

[46] A. Rossi, D. Barbosa, D. Firmani, A. Matinata and P. Merialdo, Knowledge graph embedding for link prediction: A comparative analysis, *ACM Transactions on Knowledge Discovery from Data (TKDD)* **15**(2) (2021), 1–49.

[47] M. Ankerst, M.M. Breunig, H.-P. Kriegel and J. Sander, OPTICS: Ordering points to identify the clustering structure, *ACM Sigmod record* **28**(2) (1999), 49–60.

[48] M. Nickel, V. Tresp and H.-P. Kriegel, Factorizing YAGO: scalable machine learning for linked data, in: *Proceedings of the 21st international conference on World Wide Web*, 2012, pp. 271–280.

[49] P. Ristoski, S. Faralli, S.P. Ponzetto and H. Paulheim, Large-scale taxonomy induction using entity and word embeddings, in: *Proceedings of the International Conference on Web Intelligence*, 2017, pp. 81–87.

[50] P. Ristoski and H. Paulheim, RDF2vec: RDF graph embeddings for data mining, in: *International Semantic Web Conference*, Springer, 2016, pp. 498–514.

[51] T. Mikolov, K. Chen, G. Corrado and J. Dean, Efficient estimation of word representations in vector space, *arXiv preprint arXiv:1301.3781* (2013).

[52] F. Martel and A. Zouaq, Taxonomy extraction using knowledge graph embeddings and hierarchical clustering, in: *Proceedings of the 36th Annual ACM Symposium on Applied Computing*, 2021, pp. 836–844.

[53] Z. Ding, D. Cao, L. Liu, D. Yu, H. Ma and F. Wang, A Method for Discovering Hidden Patterns of Cybersecurity Knowledge Based on Hierarchical Clustering, in: *2021 IEEE Sixth International Conference on Data Science in Cyberspace (DSC)*, IEEE, 2021, pp. 334–338.

[54] A. Hogan, E. Blomqvist, M. Cochez, C. d'Amato, G.D. Melo, C. Gutierrez, S. Kirrane, J.E.L. Gayo, R. Navigli, S. Neumaier et al., Knowledge graphs, *ACM Computing Surveys (CSUR)* **54**(4) (2021), 1–37.

[55] C. Gutierrez and J.F. Sequeda, Knowledge graphs, *Communications of the ACM* **64**(3) (2021), 96–104.

[56] D.J. MacKay, D.J. Mac Kay et al., *Information theory, inference and learning algorithms*, Cambridge university press, 2003.

[57] E. Abbe, Community detection and stochastic block models: recent developments, *The Journal of Machine Learning Research* **18**(1) (2017), 6446–6531.

[58] T.S. Ferguson, A Bayesian analysis of some nonparametric problems, *The annals of statistics* (1973), 209–230.

[59] A.-L. Barabási and R. Albert, Emergence of scaling in random networks, *science* **286**(5439) (1999), 509–512.

[60] T. Griffiths, M. Jordan, J. Tenenbaum and D. Blei, Hierarchical topic models and the nested Chinese restaurant process, *Advances in neural information processing systems* **16** (2003).

[61] D.M. Blei, T.L. Griffiths and M.I. Jordan, The nested chinese restaurant process and bayesian nonparametric inference of topic hierarchies, *Journal of the ACM (JACM)* **57**(2) (2010), 1–30.

[62] J. Sethuraman, A constructive definition of Dirichlet priors, *Statistica sinica* (1994), 639–650.

[63] J. Pitman, *Combinatorial stochastic processes: Ecole d'eté de probabilités de saint-flour xxxii-2002*, Springer, 2006.

[64] K. Toutanova and D. Chen, Observed versus latent features for knowledge base and text inference, in: *Proceedings of the 3rd workshop on continuous vector space models and their compositionality*, 2015, pp. 57–66.

[65] N. Jain, J.-C. Kalo, W.-T. Balke and R. Krestel, Do Embeddings Actually Capture Knowledge Graph Semantics?, in: *European Semantic Web Conference*, Springer, 2021, pp. 143–159.

[66] G.A. Miller, WordNet: a lexical database for English, *Communications of the ACM* **38**(11) (1995), 39–41.

[67] L. Hubert and P. Arabie, Comparing partitions, *Journal of classification* **2**(1) (1985), 193–218.

[68] C.E. Shannon, A mathematical theory of communication, *The Bell system technical journal* **27**(3) (1948), 379–423.

[69] W.M. Rand, Objective criteria for the evaluation of clustering methods, *Journal of the American Statistical association* **66**(336) (1971), 846–850.

[70] R.C. Nickerson, U. Varshney and J. Muntermann, A method for taxonomy development and its application in information systems, *European Journal of Information Systems* **22**(3) (2013), 336–359.

[71] K.D. Bailey, *Typologies and taxonomies: An introduction to classification techniques*, Vol. 102, Sage, 1994.

[72] T. Salimans, D. Kingma and M. Welling, Markov chain monte carlo and variational inference: Bridging the gap, in: *International conference on machine learning*, PMLR, 2015, pp. 1218–1226.

[73] D.M. Blei and M.I. Jordan, Variational inference for Dirichlet process mixtures, *Bayesian analysis* **1**(1) (2006), 121–143.

[74] C. Wang and D. Blei, Variational inference for the nested Chinese restaurant process, *Advances in Neural Information Processing Systems* **22** (2009).

[75] J. Chen, J. Zhu, J. Lu and S. Liu, Scalable inference for nested Chinese restaurant process topic models, *arXiv preprint arXiv:1702.07083* (2017).

[76] M. Zhou, Infinite edge partition models for overlapping community detection and link prediction, in: *Artificial intelligence and statistics*, PMLR, 2015, pp. 1135–1143.

[77] P. Rai, C. Hu, R. Henao and L. Carin, Large-scale bayesian multi-label learning via topic-based label embeddings, *Advances in neural information processing systems* **28** (2015).

[78] X. Fan, B. Li, C. Li, S. SIsson and L. Chen, Scalable deep generative relational model with high-order node dependence, *Advances in Neural Information Processing Systems* **32** (2019).

Appendix A. Probability Mass and Density Functions

In this appendix, we provide the probability mass and density functions for the distributions used in our paper. Probability mass functions capture the probability of a discrete random variable realizing a value, denoted x:

$$\text{Bernoulli}(p,q) = p^x q^{(n-x)} \tag{A.1}$$

$$\text{Multinomial}(\mathbf{p}) = \frac{\Gamma(\sum_i x_i + 1))}{\prod_i \Gamma(x_i + 1))} \prod_i^L p_i^{x_i} \tag{A.2}$$

Where p, q, and $\mathbf{p}$ are the parameters of their respective distributions. Probability density functions capture the relative likelihood of a continuous random variable realizing the value x:

$$\text{Beta}(\alpha,\beta) = \frac{x^{\alpha-1}(1-x)^{\beta-1}}{\text{B}(\alpha,\beta)}, \text{B}(\alpha,\beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)} \tag{A.3}$$

$$\text{Dirichlet}(\boldsymbol{\alpha}, L) = \frac{\Gamma(\sum_{l=1}^L \alpha_l)}{\prod_{l=1}^L \Gamma(\alpha_l)} \prod_{l=1}^L x_l^{\alpha_l - 1} \tag{A.4}$$

$$\tag{A.5}$$

Where $\alpha, \beta, \boldsymbol{\alpha}$, and L are the parameters of their respective distributions.

Appendix B. Integral Form of the Beta Function

In this appendix, we provide the derivation to obtain the integral form of the Beta function. We do this by leveraging the definition of the Beta distribution. Specifically, we begin with the identity that the integral of a probability density function with respect to its support is equal to 1 and proceed with simple integral calculus:

$$\int_0^1 \text{Beta}(\alpha,\beta)\,\mathrm{d}x = 1$$

$$\begin{aligned}
\int_0^1 \frac{x^{\alpha-1}(1-x)^{\beta-1}}{\mathrm{B}(\alpha,\beta)}\,\mathrm{d}x &= 1 \\
\frac{1}{\mathrm{B}(\alpha,\beta)}\int_0^1 x^{\alpha-1}(1-x)^{\beta-1}\,\mathrm{d}x &= 1 \\
\int_0^1 x^{\alpha-1}(1-x)^{\beta-1}\,\mathrm{d}x &= \mathrm{B}(\alpha,\beta)
\end{aligned} \tag{B.1}$$

Appendix C. Simplifying Level Likelihood

In this appendix, we provide the simplification of level likelihood by eliminating the Gamma function. Recall from Equation 26 that the level likelihood, $\mathbb{P}(\mathbf{g}_{ij*} \mid \mathbf{G}_{-(ij*)}, \mathbf{P}, \mathbf{Z}, \lambda, \eta)$, is expressed as follows:

$$\begin{aligned}
&\mathbb{P}(\mathbf{g}_{ij*} \mid \mathbf{G}_{-(ij*)}, \mathbf{P}, \mathbf{Z}, \lambda, \eta) \\
&= \prod_{g_{ijr}\in\mathbf{g}_{ij*}} \frac{\Gamma(\#_{-(ijr)}^{c_{pqr}=1} + g_{ijr} + \lambda)\Gamma(\#_{-(ijr)}^{c_{pqr}=0} + (1-g_{ijr}) + \eta)\Gamma(\#_{-(ijr)}^{c_{pqr}=1} + \#_{-(ijr)}^{c_{pqr}=0} + \lambda + \eta)}{\Gamma(\#_{-(ijr)}^{c_{pqr}=1} + \#_{-(ijr)}^{c_{pqr}=0} + 1 + \lambda + \eta)\Gamma(\#_{-(ijr)}^{c_{pqr}=1} + \lambda)\Gamma(\#_{-(ijr)}^{c_{pqr}=0} + \eta)} \\
&= \prod_{g_{ijr}\in\mathbf{g}_{ij*}} \frac{1}{\#_{-(ijr)}^{c_{pqr}=1} + \#_{-(ijr)}^{c_{pqr}=0} + \lambda + \eta} \frac{\Gamma(\#_{-(ijr)}^{c_{pqr}=1} + g_{ijr} + \lambda)\Gamma(\#_{-(ijr)}^{c_{pqr}=0} + (1-g_{ijr}) + \eta)}{\Gamma(\#_{-(ijr)}^{c_{pqr}=1} + \lambda)\Gamma(\#_{-(ijr)}^{c_{pqr}=0} + \eta)}
\end{aligned} \tag{C.1}$$

We can leverage the fact that $g_{ijr} \in \{0,1\}$ to define the second term as a piecewise function with respect to the value of g_{ijr}. Doing so allows us to cancel out terms which appear in both the numerator and denominator after expanding the Gamma function. This process leads to the following simplificaiton:

$$\frac{\Gamma(\#_{-(ijr)}^{c_{pqr}=1} + g_{ijr} + \lambda)\Gamma(\#_{-(ijr)}^{c_{pqr}=0} + (1-g_{ijr}) + \eta)}{\Gamma(\#_{-(ijr)}^{c_{pqr}=1} + \lambda)\Gamma(\#_{-(ijr)}^{c_{pqr}=0} + \eta)} = \begin{cases} \#_{-(ijr)}^{c_{pqr}=1} + \lambda & g_{ijr} = 1 \\ \#_{-(ijr)}^{c_{pqr}=0} + \eta & g_{ijr} = 0 \end{cases} \tag{C.2}$$

This allows us to put together Equations C.1 and C.2 to get the following, ass seen in Equation 26:

$$\mathbb{P}(\mathbf{g}_{ij*} \mid \mathbf{G}_{-(ij*)}, \mathbf{P}, \mathbf{Z}, \lambda, \eta) = \prod_{g_{ijr}\in\mathbf{g}_{ij*}} \frac{g_{ijr}(\#_{-(ijr)}^{c_{pqr}=1} + \lambda) + (1-g_{ijr})(b+\eta)}{\#_{-(ijr)}^{c_{pqr}=1} + b + \lambda + \eta} \tag{C.3}$$

Appendix D. Marginalizing Finite Level Memberships

In order to marginalize finite level memberships, we begin with the definition of its posterior, $\mathbb{P}(\mathbf{a}_i \mid \mathbf{A}_{-i}, \mathbf{Z}, \boldsymbol{\alpha})$, which is defined analogously to Equation 14 with the exception that the Dirichlet prior is used in the this case. Formally, we obtain the following through Bayes' rule:

$$\mathbb{P}(\mathbf{a}_i \mid \mathbf{A}_{-i}, \mathbf{Z}, \boldsymbol{\alpha}) = \frac{\mathbb{P}(\mathbf{Z} \mid \mathbf{A}, \boldsymbol{\alpha})\mathbb{P}(\mathbf{a_i} \mid \mathbf{A_{-i}}, \boldsymbol{\alpha})}{\int_{\mathbf{a}_i} \mathbb{P}(\mathbf{Z} \mid \mathbf{A}, \boldsymbol{\alpha})\mathbb{P}(\mathbf{a_i} \mid \mathbf{A_{-i}}, \boldsymbol{\alpha})\mathrm{d}\mathbf{a_i}} \tag{D.1}$$

$$= \frac{\text{Multinomial}(\mathbf{a}_i)\text{Dirichlet}(\boldsymbol{\alpha}, L)}{\int_{\mathbf{a}_i} \text{Multinomial}(\mathbf{a}_i)\text{Dirichlet}(\boldsymbol{\alpha}, L)\ \mathrm{d}\mathbf{a}_i} \tag{D.2}$$

Where $\boldsymbol{\alpha}$ is a vector of L concentration parameters for each level in the distribution, namely $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, ..., \alpha_L]$ such that $\alpha_l > 0$ and L is the finite number of levels in the hierarchy. In our marginalization, we adopt the notation from Equation 15 to indicate the number of indicators in $\mathbf{z}_{i*}$. Furthermore, we define the following vector of concentration parameters to aid in readability: $\boldsymbol{\alpha'} = [\alpha_1 + \#^{\mathbf{z}_{i*}=1}, \alpha_2 + \#^{\mathbf{z}_{i*}=2}, ..., \alpha_L + \#^{\mathbf{z}_{i*}=L}]$. With these variables in place, we can derive the Dirichlet posterior for finite level indicators:

$$
\begin{aligned}
&\mathbb{P}(\mathbf{a}_i \mid \mathbf{A}_{-i}, \mathbf{Z}_{i*}, \boldsymbol{\alpha}) \\
&= \frac{\text{Multinomial}\big(\mathbf{a}_i\big)\text{Dirichlet}(\boldsymbol{\alpha})}{\int_{\mathbf{a}_i} \text{Multinomial}\big(\mathbf{a}_i\big)\text{Dirichlet}(\boldsymbol{\alpha})\ \mathrm{d}\mathbf{a}_i} \\
&\overset{(1)}{=} \frac{\left(\frac{\Gamma(\sum_{l=1}^{L} \#^{\mathbf{z}_{i*}=l} + 1)}{\prod_{l=1}^{L} \Gamma(\#^{\mathbf{z}_{i*}=l} + 1)} \prod_{l=1}^{L} (a_i^l)^{\#^{\mathbf{z}_{i*}=l}}\right)\left(\frac{\Gamma(\sum_{l=1}^{L} \alpha_l)}{\prod_{l=1}^{L} \Gamma(\alpha_l)} \prod_{l=1}^{L} (a_i^l)^{\alpha_l - 1}\right)}{\displaystyle\int_{\mathbf{a}_i} \left(\frac{\Gamma(\sum_{l=1}^{L} \#^{\mathbf{z}_{i*}=l} + 1)}{\prod_{l=1}^{L} \Gamma(\#^{\mathbf{z}_{i*}=l} + 1)} \prod_{l=1}^{L} (a_i^l)^{\#^{\mathbf{z}_{i*}=l}}\right)\left(\frac{\Gamma(\sum_{l=1}^{L} \alpha_l)}{\prod_{l=1}^{L} \Gamma(\alpha_l)} \prod_{l=1}^{L} (a_i^l)^{\alpha_l - 1}\right) \mathrm{d}\mathbf{a}_i} \\
&\overset{(2)}{=} \frac{\left(\frac{\Gamma(\sum_{l=1}^{L} \#^{\mathbf{z}_{i*}=l} + 1)}{\prod_{l=1}^{L} \Gamma(\#^{\mathbf{z}_{i*}=l} + 1)} \frac{\prod_{l=1}^{L} \Gamma(\alpha_l)}{\Gamma(\sum_{l=1}^{L} \alpha_l)}\right)\left(\frac{\prod_{l=1}^{L} \Gamma(\alpha'_l)}{\Gamma(\sum_{l=1}^{L} \alpha'_l)} \frac{\Gamma(\sum_{l=1}^{L} \alpha'_l)}{\prod_{l=1}^{L} \Gamma(\alpha'_l)}\right) \prod_{l=1}^{L} (a_i^l)^{\alpha'_l - 1}}{\displaystyle\int_{\mathbf{a}_i} \left(\frac{\Gamma(\sum_{l=1}^{L} \#^{\mathbf{z}_{i*}=l} + 1)}{\prod_{l=1}^{L} \Gamma(\#^{\mathbf{z}_{i*}=l} + 1)} \frac{\prod_{l=1}^{L} \Gamma(\alpha_l)}{\Gamma(\sum_{l=1}^{L} \alpha_l)}\right)\left(\frac{\prod_{l=1}^{L} \Gamma(\alpha'_l)}{\Gamma(\sum_{l=1}^{L} \alpha'_l)} \frac{\Gamma(\sum_{l=1}^{L} \alpha'_l)}{\prod_{l=1}^{L} \Gamma(\alpha'_l)}\right) \prod_{l=1}^{L} (a_i^l)^{\alpha'_l - 1}\ \mathrm{d}\mathbf{a}_i} \\
&\overset{(3)}{=} \frac{\left(\frac{\Gamma(\sum_{l=1}^{L} \#^{\mathbf{z}_{i*}=l} + 1)}{\prod_{l=1}^{L} \Gamma(\#^{\mathbf{z}_{i*}=l} + 1)} \frac{\prod_{l=1}^{L} \Gamma(\alpha_l)}{\Gamma(\sum_{l=1}^{L} \alpha_l)} \frac{\prod_{l=1}^{L} \Gamma(\alpha'_l)}{\Gamma(\sum_{l=1}^{L} \alpha'_l)}\right)\text{Dirichlet}(\boldsymbol{\alpha'})}{\displaystyle\int_{\mathbf{a}_i} \left(\frac{\Gamma(\sum_{l=1}^{L} \#^{\mathbf{z}_{i*}=l} + 1)}{\prod_{l=1}^{L} \Gamma(\#^{\mathbf{z}_{i*}=l} + 1)} \frac{\prod_{l=1}^{L} \Gamma(\alpha_l)}{\Gamma(\sum_{l=1}^{L} \alpha_l)} \frac{\prod_{l=1}^{L} \Gamma(\alpha'_l)}{\Gamma(\sum_{l=1}^{L} \alpha'_l)}\right)\text{Dirichlet}(\boldsymbol{\alpha'})\ \mathrm{d}\mathbf{a}_i} \\
&\overset{(4)}{=} \frac{\left(\frac{\Gamma(\sum_{l=1}^{L} \#^{\mathbf{z}_{i*}=l} + 1)}{\prod_{l=1}^{L} \Gamma(\#^{\mathbf{z}_{i*}=l} + 1)} \frac{\prod_{l=1}^{L} \Gamma(\alpha_l)}{\Gamma(\sum_{l=1}^{L} \alpha_l)} \frac{\prod_{l=1}^{L} \Gamma(\alpha'_l)}{\Gamma(\sum_{l=1}^{L} \alpha'_l)}\right)\text{Dirichlet}(\boldsymbol{\alpha'})}{\left(\frac{\Gamma(\sum_{l=1}^{L} \#^{\mathbf{z}_{i*}=l} + 1)}{\prod_{l=1}^{L} \Gamma(\#^{\mathbf{z}_{i*}=l} + 1)} \frac{\prod_{l=1}^{L} \Gamma(\alpha_l)}{\Gamma(\sum_{l=1}^{L} \alpha_l)} \frac{\prod_{l=1}^{L} \Gamma(\alpha'_l)}{\Gamma(\sum_{l=1}^{L} \alpha'_l)}\right) \displaystyle\int_{\mathbf{a}_i} \text{Dirichlet}(\boldsymbol{\alpha'})\ \mathrm{d}\mathbf{a}_i} \\
&\overset{(5)}{=} \text{Dirichlet}(\boldsymbol{\alpha'}) \qquad \text{(D.3)}
\end{aligned}
$$

Where (1) is obtained by applying the definitions of the Multinomial and Dirichlet distributions as per Equations A.2 and A.4, respectively; (2) leverages the definition of $\boldsymbol{\alpha}'$ to group level memberships and introduces cancelling numerator and denominator terms using $\boldsymbol{\alpha}'$ to obtain a Dirichlet probability density function as shown by replacement in (3); (4) groups terms constant with respect to $\mathbf{a}_i$ in integration; and (5) leverages the law of total probability.